AGILE FRAMEWORKS®

THE BUY VS. BUILD QUESTION: Construction Materials Testing & Lab Software

Field Information Management Systems (FIMS) and Laboratory Information Management Systems (LIMS)

OVERVIEW

As a leader of an engineering firm providing construction materials testing, inspections, observations and lab services, you have or will likely need to decide whether to buy or build the business software applications that address functions such as:

- Scheduling & Dispatch
- Field Services, Data Collection, & Reporting
- Quality Control and Compliance Processes & Procedures
- Lab Services
- Report Delivery & Management
- Enterprise Content Management

This White Paper is intended to help you make the best possible Buy vs. Build decision, considering both costs and qualitative dimensions. The costs, often referred to as the **Total Cost of Ownership (TCO)** include multi-year outlays for software development or acquisition, systems hosting, support, and on-going enhancement and maintenance. We'll dive into these shortly, but first let's consider how you as a firm leader can help prepare your organization to make the most appropriate decision.

CLIENTS, CORPORATE OBJECTIVES, AND STRATEGIC CONSIDERATIONS

Clients First!

According to numerous industry studies, your clients want fast and accurate reporting in a consistent manner - no matter how many of your staff, offices, disciplines, or departments are involved. They want you to be responsive to new requirements, flexible in providing them new services, and able to grow with them – and they generally see your firm as one brand even if they call on multiple offices or disciplines. As a firm leader, you need to bring forth the needs of your clients from an enterprise perspective, and not limit such deliberations to individual division, department, or discipline needs.

Strategy Matters

Serious consideration should also be given to the long-term goals and strategy of your company as a whole. Such considerations extend beyond specific departmental concerns, and firm leaders need to champion their inclusion in any solution. Strategic requirements could include:

- Supports consistency of work product and services across lines-of-business and locations
- Solution scales to support enterprise-wide anticipated growth or seasonality
- Supports cross-functional, cross-disciplinary, and cross-departmental business processes
- Enables access by different user audiences, locations, work environments, connection modes, and devices
- Flexible enough to extend to new lines-of-business -- either developed or acquired
- Long- term maintainability and support, as growth and new capabilities evolve in particular disciplines, departments, or locations

While divisional, departmental, discipline and P&L representatives should always have a seat at the table, the considerations such as the above should be championed by firm leaders to ensure an optimum outcome for the entire firm.

Perceived Benefits of In-house Developed Systems

Even if the need for new software is not brought forward, home-grown systems have sprung up within most firms, typically by specific individuals or departments to streamline their own processes. Each comes with their own proponents, making it more difficult to advance the type of firm-wide decision framework described above. Both good intentions and creativity contribute to the proliferation of “do-it-yourself” solutions. Some of the perceived benefits of in-house developed solutions include:

It’s Cheaper.

Prevailing opinion suggests that in-house developed solutions are less costly than solutions that are licensed or subscribed. However, diluting a firm’s focus on its core competencies is a significant hidden cost - and rarely acknowledged. In fact, the most (some would say only) legitimate reason to develop an in-house solution is where nothing else is available on the market that addresses a firm’s unique needs.

“There is no doubt that Agile Frameworks MetaField® will lower the cost of doing business compared to an in house system while providing better stability, data redundancy and a constantly improving software system”

— Uri Eliahu, CEO, ENGEO

I Have More Control.

While there is no such thing as complete control, when you build your own solution you do have control over how fast the application is developed, the technology platform(s) used, how it is hosted, where data is stored, the look and feel of your reports, service levels, the product roadmap and user accessibility. That said, take note: more control generally means more cost. You are an audience of one for the systems you develop, rather than leveraging a purchased system. Over time, growing costs and investment may begin to dictate what you choose to do.

It Fits Me to a T - Today.

A custom developed solution will be tightly linked to a specific group or individuals’ way of doing business. This provides many benefits, because the solution reflects their way of delivering value. This can, however, be a double-edged sword. Custom-designed software eventually becomes a constraint on change, such as adapting to evolving best practices, new locations/ time zones, compliance requirements, or the growth and evolution of your firm. Custom- designed technology solutions tend to reinforce “that’s the way we’ve always done it” and “if it ain’t broke, don’t fix it” cultures. And, they are notoriously difficult to replace if influential firm members consider them their “baby”.

I Can Trust My People.

It’s likely you have a high level of trust with your in-house employees, and perceive less risk with them versus an external software provider, who may go out of business or be sold. This should be tempered with the recognition that employees skilled in software development are in high demand, very expensive, and can change employment at will. If you use contractors to develop your in-house software, you will generally have limited control over how their knowledge and support will be available to you in the future.

Flexibility is Free.

When you develop in-house, you can be more flexible about the commitments you make to scope, schedule, and budget. This flexibility also brings greater responsibility to manage all aspects of the development, implementation, support, and maintenance of the solution – detail activities that require time and energy from your leadership team. It’s another instance of distracting from the firm’s core competencies. Managing the roadmap also means fixing the potholes!

THE NUTS AND BOLTS – AND DOLLARS!

Armed with some insights, its' time to dive into the Total Cost of Ownership (TCO) of providing software for your business. The three major components of cost we will address are Software Development, Infrastructure and Hosting, and Maintenance and Support.

"MetaField is an enterprise solution that enables all of our offices to share the same look and feel so that we are delivering a consistent product to the customers we serve"

— Jeremy Arias, Vice President, Arias Geoprofessionals

SOFTWARE DEVELOPMENT

Developing software applications in-house takes time, money, and talent. Few engineers are also successful software developers, and the sad truth is that 2/3 of software projects still fail to meet their scope, schedule and budget objectives today. Moreover, software development is an ongoing exercise - it is never truly "done". There is always the pressure of ongoing requirement demands, the next technological leap, or competition from a more nimble competitor.

Engineering firms essentially have two options for in-house development of software applications guiding scheduling and dispatch, field data collection, quality control, lab services, digital signing and report delivery:

- **Option 1 –** Development of an enterprise-class, end-to-end, fully integrated database solution, or
- **Option 2 –** Development of a silo/discipline-driven, forms-based solution that may or may not include database integration.

While both options can deliver business benefits, Option 1 implies a much more rigorous effort, while Option 2 (while initially easier) constrains scalability, flexibility and extendability.

Option 1 – The enterprise-class, end-to-end, fully integrated database solution.

Software development at this level requires highly-skilled personnel to perform requirements definition, systems design, coding, and testing tasks. The following chart illustrates the recurring monthly cost of a nominal, team development approach. Smaller teams will extend timeframes and have more inherent risk by betting on a single/few person(s).

Staff Quantity	Role	Monthly Compensation	Monthly Extended
0.5	Program Management	$10k	$5k
1	Architect	$8k to $10k	$8k to $10k
1	Senior Developer	$6k to $10k	$6k to $10k
2 to 3	Mid-Level Developer	$4k to $7k	$8k to $21k
2	Quality Assurance	$3k to $5k	$6k to $10k
1 to 2	Business Analyst	$6k to $10k	$6k to $16k
7.5 to 9.5			$39K to $70k

It's easy to see how this approach may run into hundreds of thousands of dollars or more in a fairly quick manner. "Off-shore" software development resources can provide lower development labor costs, but this savings must be offset to some extent by the requirement for intensive, continual communication and oversight as well as the risk of intellectual property theft.

With regard to the time to develop the application(s), you should consider this to be a several year investment, with months or even years of investment before your production release. This will be followed by ongoing support and maintenance costs.

Option 2 – The silo/discipline driven, forms-based solution.

This approach can take many forms, often driven by the experience and skill set of members within a firm who are willing to "kick in" to solve the problem. Rarely are these members professional software developers. Significant benefits can be achieved, but often with both hard and soft expenses that are rarely sustainable over the long term. The following questions should be considered:

- What are the opportunity costs to the business when internal staff are focused on non-core competencies?
- Do development, support, and maintenance activities require "heroic" effort from specific individuals? Are you depending on one or two key people?
- Such solutions often require the significant interaction and interoperability of many different software components. Should the primary developers/supporters become unavailable, is such a "cobbled-together" solution understandable and maintainable by "off-the-shelf" technical resource contractors or staff?
- If more than one locally installed software component is required, what constraints does this place on the devices, modes-of-operation, and support resources?
- If the locally installed software components are operating system based, what happens when the operating systems and 1 or more components become incompatible?
- Two-thirds of the total cost of ownership of a long-term solution is often incurred under the un-glamorous areas of support and maintenance. Are the organization and the individuals involved prepared for this?

We are assuming the use of common tools such as Microsoft® Excel® and Microsoft® Word® for this option. You will need someone that is a subject matter expert in one or more disciplines with Excel / Word / Macro formula language.

As a rule of thumb, an estimated investment - for each discipline - is 4-6 months and $40,000 to $60,000 plus an estimated $10,000+ per year for ongoing maintenance. You also will require one or more administrative personnel for report creation at $35,000 - $40,000 each, per year.

INFRASTRUCTURE AND HOSTING COSTS

Infrastructure and Hosting refer to the servers, databases and data storage, network communications gear, and environmental support (power, A/C, etc.) needed to build, deploy and support a software solution. You may be planning on using your own or acquiring more infrastructure, or paying a fee(s) to a hosting solution provider to support your systems. Either way, you need to determine how much you really need. One way to gauge this is to ask and answer the question: “How much downtime can your company afford?”

In today’s competitive environment, no company can afford significant downtime. It is extremely expensive, qualitatively and quantitatively. If people cannot work on the things you are charging clients for, the impact to your revenues, coupled with increased costs to operate less effectively and to fix the issues can be significant. Therefore, you should balance the cost of downtime against the cost of providing various levels of uptime, calculated in terms of percent (%) uptime availability.

Your uptime availability goal will drive the underlying infrastructure and hosting costs. In a 365 day year, there are 8,760 hours. While “two nines” (99%) availability translates to 87.6 hours of downtime; “five nines” (99.999%) availability translates to less than six minutes of downtime annually. A good goal is “four nines” which will provide your firm with less than one hour of unscheduled downtime per year. This goal best balances uptime and the associated cost.

If you choose to build a fully functional, mission-critical “four nines” system, you are looking at investing a minimum of $8,000 to $10,000 per month for the following basics:

- Two geographically distributed hosting sites including redundant power and bandwidth
- Three servers per location
- Potentially, Data Base Management System (DBMS) software licenses that enable replication
- A part-time system administrator with very high skills in high availability application and database management
- Suitable bandwidth, Distributed Denial of Service (DDOS), firewalls, security, etc.

In the event you choose to build a less costly solution that is more forms based, using Microsoft® Excel® and Microsoft® Word® documents; along with Microsoft Office 365 hosting, the hosting and infrastructure costs may be substantially less. However, it should be noted that the path-of-least-cost may not always be the best path to choose. Remember your charge to place clients and alignment to the firm’s strategic goals in the decision process.

ONGOING SOFTWARE MAINTENANCE

Over the life of a software platform, ongoing maintenance, support and enhancements can be 2/3 of the total cost of ownership. For example, a software platform with a projected life of ten (10) years may cost $500,000 to develop. In this scenario, the total ten-year cost of ownership is estimated at $1.5 million. In this industry, a software platform that will address 90% of the requirements of a typical construction materials testing and lab services firm with multiple offices could cost $5 million initially with a ten-year cost approaching $15 million.

Maintenance, support and enhancements also require special skillsets and expertise. Internal employees will rarely stay put for such a long term on a maintenance activity. The reality then becomes employing contractors with information technology (IT) skillsets and, more importantly, with knowledge of how your custom application is built. All of this can become quite a distraction as well as a continuing drain on resources.

Of course, any Total Cost of Ownership assessment should begin with an understanding of the scope of the project – largely defined by your Operational and Feature/Functional requirements. This scope will determine development approach, platform, schedule, and budget. See the next page for an example “Buy” solution Requirements List:

“Any software system needs to have continuous development occurring to keep up operational demands, but more importantly to keep up with the swift changes in technology”

— Stuart Thompson, President, CTL Thompson

Example Requirements List	In-house	Vendor 1	Vendor 2
Operational (run-time platform) requirements			
High Availability, e.g. 99.99% uptime			
Mean time to Recovery (MTTR), e.g. 1-hour disaster recovery			
Real-time, enterprise-wide access (worldwide)			
Single database instance for the entire enterprise			
Support mobility technology, e.g. cameras, microphones, GPS, etc.			
On-line/off-line operation			
Devices/operating systems support, e.g. Windows, Apple, Android			
Support all consulting lines-of-business under single platform			
Feature/Functional requirements			
Safety Program Support Features/Functions			
Direct ERP or Project Accounting Integration			
Real-time or near real-time data integration			
Client, Project, Employee, Scope Integration			
Highly Configurable			
Time zone management			
Office and Lab configurations			
Role-based, project level security & access control			
Project Specifications Management			
Documents & Plans			
Detailed Testing/Inspection Specifications			
Work Order Management			
Work Order Status Tracking			
Calendar Management/Scheduling			
Field User Workload Management			
Field Dispatch via Text and/or Email			
DIY (Do-It-Yourself) Forms/Reports Design Environment			
Field Activity, Sample Mgmt., Lab Test Forms			
Drag-and-Drop Report Development			
Workflow engine, Calculation scripting language			
Quality Management Features			
Cross-functional, explicit, software-based quality mgmt.			
PM/PE/ Supervisor real-time dashboard			
Multiple workflow alternatives			
Digital signature/PE Seal integration			
Lab Information Management System (LIMS)			
Direct integration with field & report administration			
Sample Chain-of-Custody			
Ongoing Standards Compliance (ASTM / AASHTO)			
Materials Specification Management			
Equipment management, calibration management			
Report Generation, Consolidation, Management, & Delivery			
E-mail delivery & audit trail			
Comprehensive distribution list management			
Enterprise Content Management, e.g. non-native reports			

EXAMPLE DECISION CRITERIA TABLE

Your decision criteria about Buy Vs. Build will of course be unique, based on the objectives and strategy of your firm. Likewise, your costs of capital, depreciation methods, labor, and expense recognition should be considered in any model. The table on the following page is just one example of a "starter" model that includes general Buy Vs. Build decision criteria for any firm:

"MetaField offers complete support, larger functionality and it provides a consistent platform for our personnel across all of our offices"

— James Murphy, COO, STRATA, Inc.

Decision Criteria Example	In-house	Vendor #1	Vendor #2
Solution Acquisition/Development/Subscription Costs			
One-Time (Or Initial)			
Periodic (e.g., monthly or annually)			
Anticipated enhancements beyond primary scope			
e.g. ERP Integration			
e.g. Scheduling/Dispatch			
e.g. Additional field/lab services support			
e.g. Document management/delivery			
Hardware Costs			
End-User			
Server/Network/Infrastructure			
Third Party Licensing Costs			
End-User			
Server/Network/Infrastructure			
Implementation/Deployment/Training Costs			
Vendor Delivered			
Internal			
Ongoing Infrastructure Expenses			
Application/Database/File System Hosting			
Access/Connectivity			
Mobile/Remote Access/Connectivity			
Systems Administration/Engineering			
Vendor Delivered			
Internal			
Ongoing Solution Security/Support/Training/Upgrade Expenses			
Vendor Delivered			
Internal			
Ongoing Administrative Expenses			
Labor (e.g. word processing, QA, file mgmt, etc.)			
Materials (e.g. paper, imaging, printing, postage, etc.)			
Ongoing Miscellaneous Expenses			
Vendor Delivered			
Internal			
Total			
5-year Totals			
10-year Totals			

ENGINEERING FIRM EXECUTIVE QUOTES

URI ELIAHU, CEO, ENGEO

“We developed our own in house software system to help us manage our testing and observation program. The software was exceptionally effective and after 3 years we had expanded it to provide a multitude of other services, including workflow management, lab information management and construction SWPPP services. Unfortunately, as the software increased in size so did the management costs. In addition, though the software was tailor made for our company, as the software expanded in size it became clear that the software was not well designed for scalability.

We have found that after moving over to Agile Frameworks MetaField® software we are reaping the benefits of a shared cost for development and a product that is able to scale well as our business grows. We are expecting to have our yearly costs decrease by half while still expanding the scope of what MetaField can do compared to our in house software. Though we have had to sacrifice some of the custom features of our in house software, we have found that the Agile team constantly reviews our software update requests and helps improve MetaField accordingly. For medium to small sized businesses, there is no doubt that Agile Frameworks MetaField® will lower the cost of doing business compared to an in house system while providing better stability, data redundancy and a constantly improving software system.”

JEREMY ARIAS, VICE PRESIDENT, ARIAS GEOPROFESSIONALS

“Having been down the path of building our own construction materials testing solution, I would have to say - “why build it?” - if there is a good commercial solution on the market. Eleven years ago (2005) we decided to build our own software but that was because we could not find anything to buy. Had a product existed we probably would have gone that route and happily bought it. The support and upkeep for an in-house solution is very expensive – we had a full-time developer managing our databases and forms.

After partnering with Agile Frameworks for their MetaField® software in 2011, we get much more functionality. We don’t have to employ our own developers and we save money on administrative expenses. Another big benefit is that MetaField is an enterprise solution that enables all of our offices and disciplines to share the same look and feel so that we are delivering a consistent product to the customers we serve. In my opinion, buying is a much better idea than building.”

STUART THOMPSON, PRESIDENT, CTL THOMPSON

“We developed our own software and used it for about twelve years. We even developed a mobile application for collecting the field data in 2004-2005, although it ultimately did not function well enough to use across our operations. We had a small group of in-house software developers who built these highly customized systems under the direction of senior engineers. A few years ago we made the decision to move away from our homegrown system to Agile Frameworks. Our legacy system needed to be updated and modernized to work with contemporary technology. We were weighing the prospect of a major re-write of our own program or buying something “off the shelf”.

One of the key components of the decision to move to an “off the shelf” solution was the continuous improvement and upkeep that comes with a software platform. We found that we did not have an appetite for the cost associated with keeping professional, in-house software developers. In our experience with the homegrown system over 12 years, we found that it was not effective or efficient to have senior engineers trying to direct a software development and maintenance program. Any software system needs to have continuous development occurring to keep up operational demands, but more importantly to keep up with the swift changes in technology. In order to do this on the scale we needed, we would require good, professional software developers and not just a couple IT guys who could work up a little code. This becomes an expensive proposition. The other driving factor in moving to an “off the shelf” program is that we could get up and going quickly and we wouldn’t have to wait through the entire in-house software development cycle. We could begin realizing return quickly.”

JAMES MURPHY, COO, STRATA, INC.

“We have utilized both an in-house developed program and MetaField for our CMT data gathering and reporting. We used an in-house developer to originate and maintain our own program. Losing that resource represented our Achilles heel.

MetaField offers complete support, larger functionality and it provides a consistent platform for our personnel across all of our offices. The ability to tailor MetaField Activities allows for branding distinction in the marketplace as well as providing a tool for customizing data gathering and reporting formats for a diversity of activities. Support responsiveness has been excellent and we appreciate the focus on customer service, in line with our own client goals.”

CONCLUSION

The buy vs. build decision is unique to each firm and its associated requirements, budget, personnel, subject matter expertise and risk tolerance.

Regardless of the position any given firm takes, it is incumbent on firm leaders to create awareness of the client-oriented and firm-wide interests that should be considered. Armed with a multi-year, Total Cost of Ownership perspective, firms can consider both the near and longer term implications of building, buying, or standing pat with their business-enabling software.

Many firms have faced this question several times, maturing from no automated solution to an in-house system, and eventually to a commercially-provided solution. In our experience, most who have done so reflect on how the "Buy" answer better serves their entire company's needs. Key drivers of this choice include predictable (and usually lower) costs, reduced risk, more stable performance, improved flexibility and scalability, and more rapid innovation.

This white paper has been provided by Agile Frameworks, LLC (www.agileframeworks.com) to support AEC firm leaders dealing with "Buy vs. Build" decisions for software supporting their field- and lab-based services. You may contact Agile Frameworks at Sales@Agileframeworks.com.

Contributors: Brian Hase (Bhase@agileframeworks.com); Doug Bonestroo (Dbonestroo@agileframeworks.com); Tracie Karsjens (Tkarsjens@agileframeworks.com); and Eddie Wells (Ewells@agileframeworks.com).