PLOTBOX

To Build or Buy? Four Steps to Cemetery Software Success

Austin
Belfast
Boston
San Francisco

+1 857 301-7810
www.plotbox.io

So you've decided to modernize your organization by implementing a new cemetery management software. Now all you have to do is decide whether to buy an off-the-shelf product or to have one custom built...easy, right? Maybe not so easy! Here's a handy **four-step guide** to help you decide.

This guide will help you:

- Define system functionality
- Assess cost vs. risk
- When to build and when to buy: examples

Read closely, because the right answer isn't the same for everyone…

What’s the difference between build and buy?

- **‘Build’** is a solution where you design, build and maintain your software in-house. Whether this is done with an existing development team, hiring in a new team, or out-sourcing the work to a development company.

- **‘Buy’** is a pre-built solution. Generally, you’ll not be involved in design or build, and all maintenance, security, updates and scalability is the responsibility of the vendor.

Making the decision to build or buy

When entering into the build vs buy decision-making process, it’s important to always bear in mind that you’re looking to deliver exceptional value to your customers quickly and efficiently.

You should also try to remember that **your data will outlive your application.** The data in your system will be around long after any computer product or application, therefore it’s so important to get it right so the data can travel through time and a variety of systems.

Follow these steps to be sure you’re making the right decision, and ultimately bringing the best value to your customers.

Follow these 4 easy steps for cemetery software success:

Step 1:
Before You Start

Define your purpose in a clear and concise manner. Consider if there are any other purposes for which this data may be used? This could have a big impact on how you structure your information.

Ensure that your data follows best practice:

- *Well-defined fields* can make or break data usability.
- *Data validation rules* - what does good data in this field look like?
- *Tables and relationships* - define the relationships between data and set out tables to illustrate the relationships.
- *Sharing and multi-user access* - will it be shared or single-use?

If you don't have the expertise for this, get it. You may need to enlist the help of a consultant, but don't get it wrong!

Step 2:
Define System Functionality

Define your Minimum Acceptable System (MAS)

Firstly, what's your problem to be solved or goal to be achieved? Use this goal to determine:

- Key required users - who *absolutely* needs to have access?
- What's the must-have information needed on the database?
- What are the must-have workflows? (Tip: This is where you're going to get the best ROI)
- What's the availability of source information and how are you going to get this in a structure and format that's usable on a digital system?
- What's your *minimum acceptable return on investment*?
    - How will you pay for it?
    - How will you make money?
    - How will you save time?

Define your ideal state or desired result

This will solve a broader list of problems, pains, and aspirations, i.e. it’s an ideal state of results. Create a list of:

- Users who could benefit from the system
- Desired information to be included on the system
- Desired workflows the system will need to carry out
- Potential integrations - how you want to share this information amongst other software systems
- The success metrics, by feature
- Other areas where a return on investment can be achieved if you expand the project

Validate and quantify ROI

Do this for both MAS and desired state. Below is a list of examples of areas where you might be able to find your ROI. Not all ROI will be measured in hard dollars! It’s also not an exhaustive list - it’s important to do your own research to find out what would be the most beneficial measurements for your own cemetery:

- Increase sales
- Increase transaction value
- Reduce labor
- Increase research efficiency
- Improve transaction processing
- Control maintenance expenses
- Improve supervision
- Improve communication
- Reduce materials or inventory costs
- Improve accuracy
- Improve family service
- Build brand and improve image
- Reduce risk

Quantify and validate costs

After quantifying ROI, you'll want to figure out what your costs are going to be. Again, some examples of costs might be:

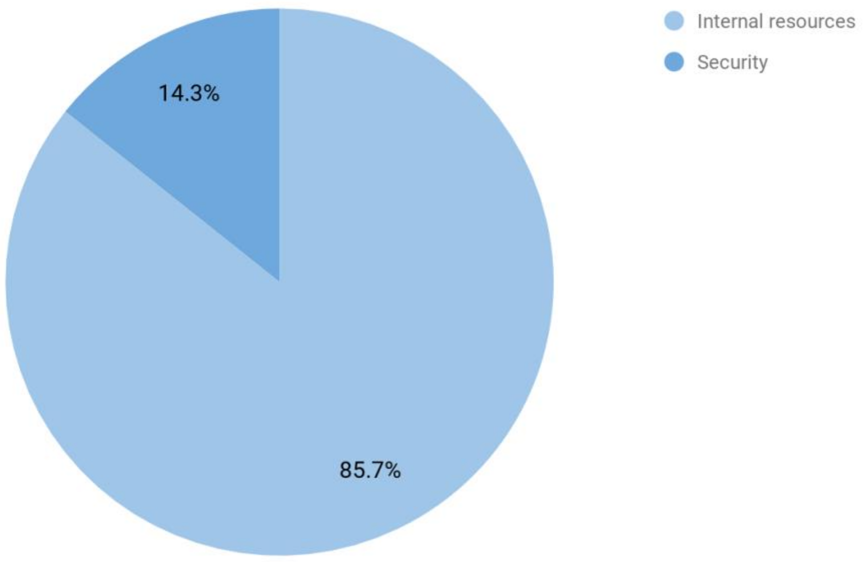
For the purchased system

- License costs
- Hosting and or hardware
- Upgrades
- Project management (internal)
- Implementation
- Training
- Support
- Security

For internal development

- Project management
- Hosting, licensing and/or hardware
- Development and implementation
- Quality assurance
- Changes and change management
- Training
- Support
- Security
- Succession

Now that you've considered what your costs might be, you might have a better idea of what would be your biggest challenge in completing a build or buy a project. In a recent poll of cemetery professionals, the biggest challenges were perceived to be internal resources (85.7%) and security (14.3%). However, no respondents answered 'availability of data' or 'ongoing support' as being one of their biggest challenges.

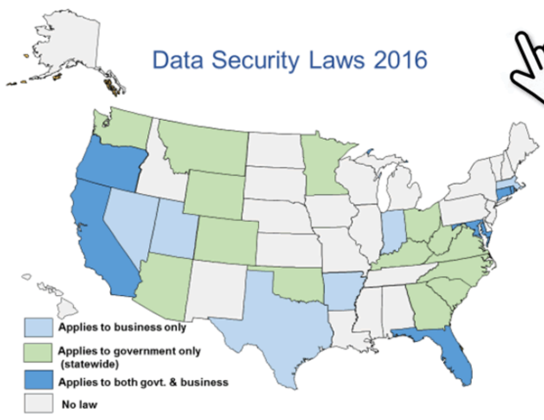


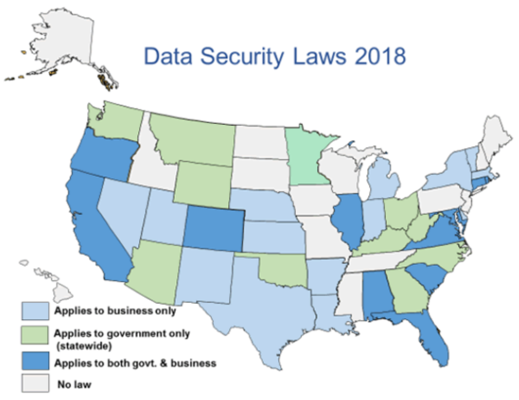
Step 3:
Cost and Risk

Security and privacy: a constantly moving target

Here's the challenge:

- The number of states with data security and privacy laws has doubled from 2016 to 2018
- Even the states who already had these laws, they've tightened up their laws to include more and more people
- The California Consumer Protection Act is due to take effect in January 2020 - it's very rich in requirements in terms of how you handle the security of your data
- The European GDPR is an act which protects the data of **all** European citizens


Data Security Laws 2016
Applies to business only
Applies to government only (statewide)
Applies to both govt. & business
No law


Data Security Laws 2018
Applies to business only
Applies to government only (statewide)
Applies to both govt. & business
No law

Source: National Conference of State Legislatures

The problem is, there will be situations when your software is no longer compliant with current regulations, and you'll have to make changes. You have to think about:

- If you're developing your own system, do you have the resources to maintain compliance?
- If you're buying, some of the smaller vendors might not have enough of their own resources to maintain compliance or have the appropriate privacy capabilities.
- You need to get penetration tested. Do you have the resources to get this done on a regular basis?

Myths busted

- We only have dead people on our cemetery, therefore data protection acts don't apply to us

You'll still have living people on your system, whether that be plot owner, next of kin, deed owners, etc. If you're selling plots you'll probably have alive people on your system. Even your own staff are covered under most data protection acts! The easiest way to deal with it is to look at an existing software with a robust system to deal with security.

- GDPR doesn't apply to cemeteries in the US

Lots of cemeteries in the US, especially in the north-western areas, are highly likely to have European citizens in their database.

- To keep my system secure, I'll keep it on-premises

Is this really more secure? Not if you don't have an on-site team looking after it - in fact, you're probably at a greater risk. Do you have the resources to look after this constantly? It also depends on the vendor; some systems just don't have good security.

Workflows

What are workflows? They're the execution of process steps with data validation to replace manual process steps. Some things to remember about workflows are:

- They're usually the biggest source of ROI on projects.
- The development of workflows can take 5 X the time of setting up the data sources they act on.
- Quality assurance increases exponentially with the number of processes you add.
- Workflows change and you have to be able to respond to that - it can be very expensive to remain static and stick with the way you've always done things!
- Successful workflow implementation = improved ROI.

Can you define and execute on workflows?

No - you might want to look at a pre-built system.
Yes - you might be successful in building a system yourself.

User experience

It's not a solution if nobody uses it…

Users are really reluctant to use a poorly designed system. An easy-to-use system generates user and feature adoption. So how do you do that?

- Software and internet best practices - do you have the experience and knowledge to adopt those in your system? To ensure individuals aren't going to stop when they come up against the first problem.
- Industry best practices and workflows. If you have the ability to understand them and the desire to adopt them, then you'll be able to add them into your system. Alternatively, there'll be systems with these workflows already build in.
- User role configuration - does your software have the ability to present users with only the information that they need?
- Documentation - both within the software and user manuals. This can be expensive but is required so staff will use the system.

Failure to adopt = no or negative ROI

Will your staff use it? Can new staff get started easily?

Support

This is the most underestimated of all the requirements when looking at build projects! You'll need to be able to provide support for:

- Users who have general ongoing operation questions
- Training for staff changes
- Response to external change
- Infrequently used features

Diminishing usage = diminishing or negative ROI.

Who will answer questions and resolve problems? Do you have the budget to handle it?

Integrations

You'll need to consider if your data needs to be shared with other applications. A lot of people will think it's fine to type the same information into multiple applications which has its own issues. Every application is a moving target - they'll change over time and you'll need to be able to adapt to those changes. Some things to remember include:

- Have good data and process validation rules built into your software
- Auditability - can I verify that a process was done correctly?
- APIs - Application Program Interfaces. These allow the interchange of data between two pieces of software. They'll often bypass the data validation rules and workflows built into your system. So bad data can easily get fed between one system and another. However, APIs can work when done correctly and are actively monitored.
- You can also move information from one software to another through importing and exporting. This method will mean you have an audit log, however, will often require an additional, very manual step.

However, there is another option:

- Choosing a broad-based application that can be a single source of truth.
- Can you support required integrations? Are they necessary? Are there any alternatives?

The never-ending project

Roughly 70% of software development projects are abandoned before going live and this is due to:

- Time
- Poor management
- Poor definition of the project

So how can you minimize this risk?

- Establish absolute clarity on Minimum Acceptable System (MAS)
- Produce detailed definition, scoping, and resource allocation for MAS
- Evaluate ROI for MAS, quantified and compared to resource requirements
- Employ professional project management (not by a developer) with performance milestones (it may be worth considering getting a third-party PM)
- Provide the same analysis for all potential enhancements
- Prevent scope creep
- Can you get your project over the line?

Succession

If your project leader, developer, or sponsor was incapacitated, who would take over? And what kind of process do you have to handle that transition? Some things you can do to protect that include:

- Document your designs
- Workflow documentation - including why are you taking these steps or doing this calculation?
- Security documentation
- Successor handoff plan and execution
- Maintain documentation over time

Projects with no succession plan eventually die sad and expensive deaths! Confirm succession resources and costs before you start.

Mistake 13:
When to Buy and When to Build: Examples

When to build

Unique competitive advantage

If you have a unique competitive advantage that you can express in software, for example, calculators, information aggregators, or similar systems. If it's unavailable commercially and you don't want to share your secrets with an external developer, then you may want to consider building. Some tips are:

- Stick to the solution and keep it simple
- Don't focus on integrations; manual entry of key information still yields benefit (will start to get data leakages and diminish your competitive advantage)
- Keep it private, and limit use to key beneficiaries
- Protect with encryption, digital watermarks, intellectual property registration, NDAs, etc.

When to build (continued)

Small records management project

If you're dealing with simple data structures; minimal workflow development; limited users with limited function; and have the resources to perform data entry then you might want to consider building your own software. Because it won't use a lot of resources, you'll see some quick ROIs:

- Can reduce retrieval times and limit data loss
- Limited scope, but limited ROI from client service and risk mitigation
- Minimal downside risk

Large co-dependent internal systems

You might want to consider building your own system if:

- You have significant other internally developed systems that will depend on or share data with this system
- You have defined the scope of the project and its interrelationships in complete detail
- You have the internal resources and executive support to execute the initial project
- You have the internal resources to manage ongoing support, quality assurance, scope change, integration, and staff training needs
- You have the depth of staff to assure its succession
- Taking into account all costs, you are still able to demonstrate positive ROI
- The ROI exceeds that of a purchased system
- If you've really considered all of the costs. Will this be a never-ending project?

Considerations for buying a software

Consider buying a ready-made software solution if:

- There’s already an off-the-shelf solution available
- Year-over-year predictability of expense is desired
- Limited internal resources or expertise is available
- You have the desire to adopt industry and software best practices
- You need to realize a timely return on investment
- There are concerns over usability or user adoption
- The solution includes workflows or business processes
- The complexity of the project exceeds available resources
- Maintenance resources would consume any return on investment

Buy it: configurable vs. customized systems

- Configuration allows for selection of options within the core system.
- Configurable systems allow simple change over time, but may not address specific needs.
- Customization involves altering the core system for specific needs (typical lifecycle of a customized system is 3 to 7 years).
- Customized systems can cost less and have more features than internally built systems but usually become frozen in time, with expenses for change similar to "built" systems and become very difficult for the developer to support.

The alternative to build or buy

Downscale

- Identify the smallest capability that represents the greatest result
- Redefine your Minimum Acceptable System

Example: Is it a document management or records management problem?

Start small and grow into it

- This option carries the biggest risk of failure
- It'll fail if the initial design doesn't accommodate expansion or if additional resources are unavailable
- It also carries the risk of initial data being unusable upon expansion

Most "start small" projects stay small. Some are abandoned completely and replaced by larger systems. Few have continued development.

Do nothing!

We're not joking when we say that sometimes the best option is to do nothing! This is an option to consider when:

- The problem isn't big enough to invest resources
- The resources to convert data aren't available
- Staff or potential user support is lacking (best to try and evaluate this)
- Cost of conversion is too high to achieve ROI

This is the easiest way to avoid a failed project!

Conclusion

This is a rough guide for helping you to decide whether to buy an off-the-shelf cemetery management solution or develop one for yourself. Don't forget to **customize your build vs buy assessment process** to meet your cemetery's needs.

If an organization has an in-house development team, there's sometimes pressure to build for various reasons. However, do consider that it's usually **far cheaper and faster to buy than to build**. After all, if a problem has already been solved in a commercial product, why solve it again?

To find out more information or to arrange a PlotBox demo, please contact us on info@plotbox.io

info@plotbox.io
www.plotbox.io