Making Sustainable Technology Choices

TOM SCHEINFELDT

Whether it’s implementing a new content management system for your website, using a new project management platform to plan an upcoming exhibition, or rolling out a new customer relationship management package, making technology changes requires meaningful institutional investments. Because new technologies have significant costs—sometimes in license fees, but always in staff time and training—cultural heritage organizations are understandably reluctant to adopt them too often. Moreover, when it does become necessary to make a change, organizations want to make very sure that the new technologies they choose will have staying power—that they won’t incur ongoing maintenance costs or become obsolete in a few short years.

As someone who both administers a public humanities organization and has built software packages for cultural heritage organizations, I am often asked how to pick a new technology platform. Implementing new technology always requires experimentation, so there are no hard-and-fast rules. There are, however, certain guidelines that can make the process more productive and sustainable.

MANAGE EXPECTATIONS

When picking a new software platform, it is vital to start with clear-eyed expectations for what and how much the organization will get out of the software, in terms of both increased capacity and longevity.

Most new technologies are initially sold as time savers, but rarely do they end up saving us time. It is important to understand that a new piece of technology will generally not help an organization save staff time, even after its initial installation is complete. Instead, new technologies increase our productivity. We still have as much work as ever, we are just able to do more of it and do it more effectively.
The introduction of the washing machine in the early 20th century, for example, did not immediately allow women in the home more leisure time, though this is what its marketing promised. Instead, in many cases, women using early washing machines actually spent more time on laundry. Rather than saving women time, the washing machine allowed families to have more and cleaner clothes.

Likewise, do not expect the adoption of, for example, a new customer relationship management platform or web content management system to save you or your staff time. It follows that technology will not allow you to “downsize,” as it has in some sectors of the economy, particularly manufacturing. Instead, you should expect that new technologies will make your marketing or outreach efforts better. You may be able to keep in contact with donors and patrons more regularly, reach audiences over greater distances, distribute educational materials more widely, and tell more compelling stories. But you will still need staff to keep these systems and their contents up to date, to experiment with new methods and strategies, and to follow up with all the new people.

Young children interact with simulated water, logs, alligators, and birds in River of Grass, an exhibition based on the Florida Everglades ecosystem at the Frost Museum of Science in Miami.

PHOTO BY ROBERTSON G. ADAMS LICENSED UNDER CC BY-SA 4.0
you’ve reached. In the essentially people-centered business of cultural heritage, technology is for enhancing the capabilities of our people, not for replacing their jobs.

Organizational decision-makers should also make technology decisions with the clear-eyed expectation that, whatever technology they choose, it will likely be obsolete in less than five years. I know this is hard to stomach, but irresistible forces are at work here.

To a great extent, obsolescence is simply a function of how digital technologies develop. Amazingly, more than 50 years after it was posited, Moore’s Law—the prediction by Intel co-founder Gordon Moore that computing power should double approximately every two years—still more or less holds. Hardware companies use this geometric increase in computing power to maintain aggressive product cycles: the computers and mobile devices on the shelves in two years will be twice as fast as the ones you buy today. And your computer is designed to be old by then.

Now maybe you don’t care about having the fastest laptop or latest smartphone. Unfortunately, it’s not just about hardware. Software developers design their products to run on the latest hardware—all the more so in these days of automatic updates via mobile app stores like the iOS App Store and Google Play and subscription software services such as Office 365. Therefore, roughly two years from now, whether or not you will want a faster device, the software delivered to it will be designed to run on something twice as powerful as the one you have today. Snapchat’s iPhone app is 51 times larger in 2017 than it was in 2013.

Alas, mobile phone operators are in on it too: this is the reason for two-year contracts and financing plans. Verizon and AT&T know that you’ll be fed up with your phone in two years when it slows to a crawl trying the run the latest update of the Facebook mobile app. The planned obsolescence perpetrated by hardware makers, software companies, and mobile operators may be lamentable, but it has been a fact of computing for more than a half century. We shouldn’t expect it change. What we should do is plan, budget, and fundraise accordingly.
START LOW TECH

When picking a new technology platform, the strong temptation is
to go for the option with the most features. But choosing software
is a lot like choosing a car. A Chevrolet and a Mercedes-Benz are
equally capable of getting you to work. The Benz may do it in more
comfort and style, and it will certainly have additional features to
help you get there—a navigation system or adaptive cruise control,
perhaps. But the Chevy will do the essential work just as well, and it
will leave you with more resources to use elsewhere.

Likewise, when choosing a new technology for your organization,
start with the lowest-tech solution possible, and only move up to a
higher-tech solution when you reach the limitations of the original
product. For example, in picking a collaborative project management
tool for your team, don’t go straight to Microsoft Project or Jira.
Start, perhaps, with a Google Doc, which offers most of the essential
features of those more sophisticated platforms—collaborative
authoring, checklists, revision histories, and user management and
tracking—without any upfront cost or setup time and with a
minimal learning curve. You could easily spend hundreds of dollars
and days, if not weeks, of staff time installing and configuring
something like Microsoft Project, or even a “freemium” online
solution like Basecamp, and be no better placed to share task lists
and milestones with your team than if you had started a Google
Doc. You can always transfer the plain text of a Google Doc or the
structured data of a Google Sheet or Excel spreadsheet to a more
sophisticated system if you find you need the additional features.
Moreover, after starting small and exhausting the capacity of a
lower-tech system, you will discover what features you really need
by repeatedly bumping up against their absence. For instance,
maybe you’ll find that you need a more “granular” permissions
structure that allows some people to view and edit some data
and others to view and edit other data. Maybe you’ll need a more
thorough notifications system to keep team members on task or
a better calendar feature.

This rule applies across the cultural heritage technology
spectrum, from project management software to web content
management to collections management. Resist the temptation to
buy the Benz—start with the Chevy. The skills that you learn while driving the Chevy will be entirely transferable to the luxury car, and the experience of driving it will help you know which options you’re looking for in a Benz.

**CHOOSE OPEN SOURCE AND OPEN STANDARDS**

Beginning about 15 years ago, in the aftermath of the dot-com boom, digitally minded cultural heritage professionals became increasingly interested in the potential of open source software—that is, software whose code is released under open copyright terms. Although it comes in many flavors, open source software is contrasted with proprietary software, which is licensed to users, usually for payment, under terms that restrict its use and redistribution. Proprietary software is “closed” in that users cannot alter or reuse its source code. Open source software, by contrast, allows users to inspect the source code, change it to reflect their particular needs, and redistribute those changes.

Note that the terms “open source” and “proprietary” refer to how software is licensed, not to whether it has been developed by a nonprofit organization versus a corporation. Plenty of corporations build open source software: Google is probably the largest, but IBM and even big proprietary software makers like Microsoft release open source software as well. Conversely, nonprofits routinely build and sell proprietary software.

And while educational and cultural organizations’ shift toward open source in the early 2000s partly stemmed from a desire to align themselves with technology communities motivated less by the corporate excesses of Silicon Valley and venture capital and more by the shared values of knowledge creation and dissemination and public service, there were also important practical reasons for choosing open source technologies in the wake of the dot-com bust. The very present realization of the fragility of corporate technology firms drove many cultural heritage and educational organizations, for whom sustainability is a core value, toward open source as the most practical, most prudent choice. What is important for our discussion is not whether one kind of software
is made by idealists and the other by corporate villains. Instead, we should evaluate what an open source software package enables you to do and what it offers in terms of sustainability—and many factors make open source a more sustainable choice.

Users can not only view open source code and make changes to it but they are also encouraged—sometimes required—to release those changes back to fellow users and developers, who can use the altered code and develop it further. This dynamic creates a virtuous cycle of continuous improvement, particularly when it comes to squashing bugs and fixing security vulnerabilities. As programmer and open source advocate Eric Raymond famously noted, “With many eyes, all bugs are shallow.” Open source software is continuously updated by a loose community of developers, and new releases are issued when they are needed, not when a copyright holder determines it will be best for business. This means that, in theory at least, your organization can always have the most up-to-date software.

The same communities that find bugs and contribute security patches, device drivers, and maintenance updates are also available for support and training. Robust open source software projects have active user forums where questions are answered quickly by fellow users who often have faced the same real-world problems. It is in the community’s own interest to produce and make freely available good documentation, instead of keeping expertise in the domain of consultants employed by software proprietors. Most successful communities host their own conferences and workshops.

And community developers are not simply writing bug fixes and driver updates for the core software—they are extending it to do new things. Most open source software packages offer extensible architectures that allow you to “plug in” new features and functionality as you need it. In thriving open source projects, many of the features you might need will be available in the form of a plug-in developed by a fellow community member. Users of open source web browsers such as Firefox and Google Chrome (which is based on the open source Chromium project) will recognize this plug-in concept from the wide range of available “extensions” for their browsers. Of course, you are not confined to
being a passive user of the software and its plug-ins—you too can be an active developer, building plug-ins to accomplish your goals and giving them back to the community.

Finally, open source software packages typically implement open data standards, which will allow you to move your data to a new platform when necessary. It may be in a proprietary software maker’s best interest to make it hard to leave—to “lock” you in by formatting your data in a way that’s only readable by its own products. An open source community has no such incentive. It’s interest, like yours, is in data portability, and its open data standards support this.

These qualities of open source software all point in the same direction: robust open source projects have active communities behind them committed to keeping the technology alive. Like all digital technology, open source software is subject to Moore’s Law and turns over on roughly a two- to three-year cycle. You’ll have to turn over with it. Doing so will necessitate becoming a member of the open source community, and if that seems like a lot of work, it is. Although you may not have to pay a license fee, running open source software is patently not “free” of cost. Rather, advocates joke that open source software is free like a puppy: you can have it, but you’re committing to a lifetime of care and feeding. Dog owners, and open source software community members, know that it’s worth it.

**INVEST IN PEOPLE, NOT TECHNOLOGY**

By now it should be clear that sustainable technology requires the time, attention, and care of people working in community, and that brings me to the most salient advice I have to offer organizations looking for a sustainable technology strategy. When making technology choices, it is best not to invest in technologies themselves, but in the people who will use and maintain them for your organization. Laying out cash on vendors and consulting is always a short-term strategy. On the other hand, making steady, consistent investments in finding people comfortable with experimentation, and then in supporting them as they explore and adopt new technologies, will create a sustainable culture of digital know-how and innovation that will serve your organization across
the unrelenting cycles of technological change. You need not hire purebred geeks or even dedicated IT staff. Empower your existing staff, support their professional development through workshops and training, and give them time to push their limits. If you’re hiring, don’t hire for particular tech skills or knowledge of a particular software platform, but with these culture-building values in mind: a demonstrated track record of experimentation with technology, a do-it-yourself approach, and a commitment to engaging with technology communities. Let your people work through the processes of migrating data to a new platform when an old one becomes untenable, starting low tech and working up to the best-fit solution, or participating in an open source community—and they will build a technology strategy that’s right for your organization.

Investing in people is costly, no doubt, but building technology know-how—and a culture of technology know-how—is your best bet for weathering the change inherent to the technology landscape. Your technology roadmap will be more crooked, and your timeline to implementation will be longer—but so will your timeline to obsolescence. FJ

TOM SCHEINFELDT is associate professor of digital humanities and director of Greenhouse Studios at the University of Connecticut.

TAKEAWAY
Read about the Greenhouse Studios design process.

TAKEAWAY
Read about the collaborative culture at Greenhouse Studios.