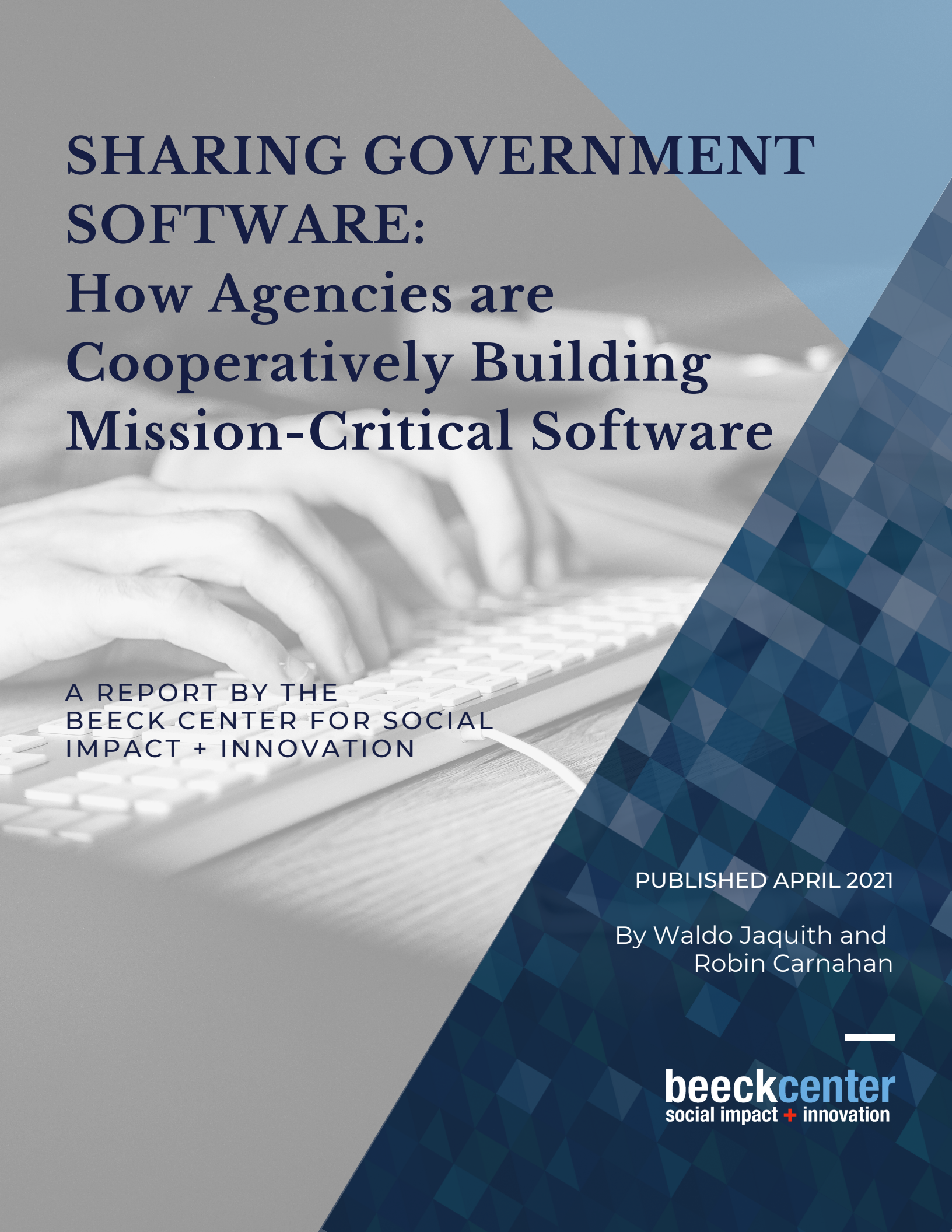

SHARING GOVERNMENT SOFTWARE: How Agencies are Cooperatively Building Mission-Critical Software

A REPORT BY THE BEECK CENTER FOR SOCIAL IMPACT + INNOVATION

PUBLISHED APRIL 2021

By Waldo Jaquith and Robin Carnahan

beeckcenter
social impact + innovation

ABOUT THE BEECK CENTER FOR SOCIAL IMPACT + INNOVATION

The Beeck Center is an experiential hub at Georgetown University that trains students and incubates scalable, leading edge ideas for social change. We believe impact at scale requires the courage to think and behave differently. Our work centers on investing in outcomes for individuals and society. We equip future global leaders with the mindset to promote outcome-driven solutions, using the tools of design, data, technology, and innovation. We convene actors across the public, private, and civic sectors to advance new tools, frameworks, and approaches necessary to achieve these outcomes.

ABOUT THE STATE SOFTWARE COLLABORATIVE

The Beeck Center's State Software Collaborative is bringing together U.S. software cooperatives to facilitate their development of shared software and services. Instead of 50 states buying 50 versions of near-identical, overpriced software, we're facilitating the cooperative development of high-quality, fair-priced software to be shared among agencies.

ABOUT THIS REPORT

This report was released April 2021 under a Creative Commons Attribution-ShareAlike 4.0 license, and should be cited as: Waldo Jaquith and Carnahan, Robin. Sharing Government Software: How Agencies Are Cooperatively Building Mission-Critical Software. Beeck Center for Social Impact + Innovation, Georgetown University, 2021.

Table of Contents - Outline Summary

EXECUTIVE SUMMARY

Since the 1960s, intergovernmental software cooperatives have quietly underpinned and facilitated the operations of government throughout the United States. These organizations are made up of two or more agencies, jointly supporting the development of software for their collective use, operating under some kind of a governance structure. Today there are many dozens of intergovernmental software collaboratives providing the software that operate DMVs, highway departments, libraries, labor agencies, insurance commissions, and transit agencies, for example. These are often housed at long-standing non-profit organizations that coordinate the interests of these agencies. Co-ops are attractive to agencies because of their break-even low costs and their software's low risk of failure, compared to custom software development.

Collectively, co-ops' budgets are hundreds of millions of dollars annually, but individually some of them operate successfully on shoestring budgets that are otherwise unheard of in government. Some of their services are so crucial that, were they to disappear today, state agencies across the nation would be brought to their knees. Paradoxically, co-ops are so little known that few state CIOs could name more than a couple of them.

Successful co-ops tend to have a clear governance structure, deliver value to users incrementally, and focus relentlessly on user needs. New co-ops should start by identifying a shared need, seek to solve a small problem, operate under a clear governance structure, architect software to suit the needs of the members, and work in the open.

Software cooperatives have a track record of reliably building and maintaining essential technical infrastructure for government. Private-sector grant makers should consider funding these organizations to improve government service delivery. Before requesting funding for major software procurements, agencies should determine whether there are existing co-ops that could solve the problem. When funding major software procurements, appropriators should consider requiring agencies to develop and share that software with peer agencies.

Introduction

Since the 1960s, agencies at all levels of government have created software to fulfill their missions, and shared that software with other agencies. By working together, these intergovernmental software cooperatives have quietly created and maintained vital digital tools that agencies rely on to serve the public. States' DMVs, highway departments, libraries, labor agencies, emergency managers, insurance commissioners, and transit agencies are all likely to rely on software created by an intergovernmental software cooperative.

Within the United States, similar municipal and state agencies tend to have approximately an 80 percent overlap in their software needs. This is because much of the function of state and local governments is the same, and research confirms that software needs tend to map to those common functions. Sharing software is a logical way to reduce the time, risks, and costs associated with major technical procurements. There are several ways that software is shared across agencies: sometimes via vendors (who develop custom software for their first agency customer, and then resell it as commercial software to subsequent agency customers); sometimes informally, by publishing under an open source license; and sometimes explicitly, via an intergovernmental software cooperative model.

Today there are dozens of software cooperatives in the United States, and surely many more that we have not yet identified. Most of these cooperatives were created independently of each other, and yet have striking similarities, despite having no published best practices or industry norms to rely on. Although their governance structures vary enormously, they largely share commitments to the incremental delivery of upgrades and to solving user needs. "Software cooperatives" is a descriptor used for the purposes of this report; these organizations do not identify as such, but instead as participants in their governmental sector (transportation, unemployment insurance, taxation, etc.).

This report reviews the features of intergovernmental software cooperatives, examines several different examples, looks at different categories of cooperatives and their governance structures, and inventories known cooperatives both within and outside of the United States. Agencies rethinking how they obtain technical functionality, budget officials looking to control costs and outcomes, or private funders that want to improve public services may find this report particularly useful.

Overview of Cooperatives

WHAT THEY ARE

An intergovernmental software cooperative is made up of two or more government agencies jointly supporting the development of software for their collective use, operating under some kind of a governance structure.

These agencies might be local, regional, state, or federal. The software might be shared between agencies within the same state or federal government, or they might be shared across states or countries. Agency staff might all directly contribute to the creation and maintenance of that software, or they might outsource the work to a vendor. The agencies might collaborate on the ongoing development of the software via an open source development model, or they might simply share compiled software. The software might be independently run by each agency, or they might collectively share a Software as a Service (SaaS) model. There are a handful of different types of cooperative sharing models (see Appendix A), but in the end, they all do roughly the same thing: develop software for the collective use of their members.

There are many cooperative-adjacent models that are outside of the scope of this report. For instance, when a central governments (e.g., a state) purchases a license for commerical software that allows for reuse by its members (e.g., counties), that is simply bulk purchasing. Another example is when an agency builds custom software and publishes the source code for anybody else to reuse independently—that is simply sharing.

IN PRACTICE: Evergreen

Evergreen is an open source integrated library system created by the Georgia Library Service in 2005. Georgia released the software under an open source license, and its subsequent popularity has led to its use in managing the collections of more than 2,000 libraries around the world. A non-profit organization, the Evergreen Project, was created to house the software, and it has a board of representatives from organizations that rely on the software. Software development is largely done by library employees and software developers contracted by libraries, although the newly created Evergreen Community Development Initiative (ECDI) is pooling members' funds to contract for software development for their collective benefit. Some of the members of ECDI are, themselves, cooperatives, making it a sort of a cooperative of cooperatives.

https://softwarecollaborative.org/cooperatives/evergreen

WHY THEY ARE USEFUL

There are three ways in which cooperatives are particularly valuable:

- **They reduce individual spending.** The price of building custom software for a single agency is only slightly less than the price of building custom software that will work for two agencies. The marginal cost decreases with each additional agency.
- **They propagate best practices.** Software is the codification of practices and workflows, and turns out to be a great mechanism for agencies to collaborate, sharing what they have learned over decades of work and making their experiences available to others.
- **They increase the odds of success.** It makes sense to implement software that has already been successfully implemented at a similar agency. This is conceptually what commercial software vendors offer, but at a high cost to taxpayers.

WHERE THEY COME FROM

Nearly every co-op began with a common user need that existed at a small number of agencies. They had the same problem at the same time, and decided to work together to solve it.

IN PRACTICE: Notify

Notify is an open source software-as-a-service tool, hosted by a central authority, that participating agencies can use to send emails, text messages, and postal letters. It was created by the United Kingdom's Government Digital Service, which uses Notify to provide communication services to national and local government agencies. Notify has been replicated by Canada's and Australia's national governments, which employ the tool in similar manners, and by the U.S. Department of Veterans Affairs.

https://softwarecollaborative.org/cooperatives/notify

Many co-ops were created from the top down, emerging from an existing interagency organization. The American Association of State Highway and Transportation Officials, for example, has been around for more than a century, but it wasn't until 1985 that they addressed the common technical needs of their members by getting into the software business. These existing interagency non-profit organizations are fertile grounds for collaborative software development, thanks to their shared mission and existing governance structure. They often start by sharing standards, move onto sharing data, and then transition into sharing software.

Other co-ops arose from less deliberate processes. For example, the state of Georgia created the Public Information Network for Electronic Services (PINES) to run the state's libraries before deciding to open source it, which resulted in Evergreen, now run by a non-profit organization and used in more than 2,000 libraries.

These sorts of co-ops don't originate in conference working sessions, but in the hallway conversations between sessions. They start with the minimum viable product, and grow organically, often over many years, even decades. They start at the bottom and grow out and up.

WHY THERE AREN'T MORE

Given the value and success of intergovernmental software cooperatives, why aren't there more of them? There are a handful of factors that appear to contribute to their relative scarcity.

Governments' budgeting and procurement processes appear to be a significant limitation on the creation and expansion of cooperatives. When a new policy is being implemented or an agency requires new technical functionality, they often begin by publishing a Request for Information (RFI), to solicit feedback from the software industry about what options exist and an approximate cost for those products or services. Monitoring RFIs and replying to them is a significant amount of work, an investment that makes sense for a software vendor, but that does not make sense for cooperatives. Agencies generally use RFI responses to make a budget request to their legislature, for example, which results in an allocation of funding to the agency. The agency then publishes a request for proposals (RFP), which is met by detailed proposals from vendors in the software industry. RFPs can be hundreds of pages long, and require proposals that are equally lengthy; again, cooperatives have neither the capacity nor the interest to write such proposals. The entire process by which agencies request funding for and acquire new technical functionality is built for purchasing software and services from commercial vendors, not for sharing and reusing software from co-ops.

Agencies are not pushed by funders to seek out co-ops because budget staff—at agency, executive, and legislative levels—are seldom aware of the existence of co-ops. If they knew about co-ops, they could encourage or require agencies to explore that option prior to requesting funding.

Even if an agency was aware of an existing co-op that they wanted to join, they might find it challenging to do. Governments have comfortable, familiar processes around budgeting and procurement; an agency wishing to join a multi-state compact or sign a memorandum of understanding with a non-profit organization would find itself far off that beaten path, faced with government attorneys who are incentivized to guide agencies away from anything new or different, in order to reduce risk.

Finally, cooperatives are simply not well known. They receive little attention, the software that they maintain tends not to be public-facing, and their low dollar values mean that they're not on a budget staff's radar. Agencies are unlikely to be aware of cooperatives as a concept, and so they are left relying on standard procurement processes.

How Cooperatives Succeed (And Fail)

SUCCESSFUL CO-OPS

Successful co-ops generally have three traits in common:

- **A clear governance structure.** This does not mean that they have a formal governance structure, but instead that all participants know what they owe to each other: they know what they will provide to the effort, what benefit they will receive, and how much control they have over the work.
- **Incremental delivery.** Instead of working for many years, waiting to release software when it's "done," these co-ops release updates to their software early and often.
- **A relentless focus on user needs.** These co-ops base all software development on what the software's users need, and primarily concern themselves with whether they have successfully addressed those needs.

The latter two—incremental delivery and focusing on user needs—form a mutually reinforcing pair that is known as "Agile software development." Many co-ops have worked like this prior to the creation of Agile, while others working in this manner may be unaware that they are practicing Agile.

UNSUCCESSFUL CO-OPS

Unsuccessful co-ops are more difficult to study, by virtue of no longer existing. While unsuccessful co-ops generally lack the traits of successful co-ops, the strongest theme tied to failure is the lack of a clear governance structure.

Governance problems are well illustrated by the Internet Unemployment System (branded as "iUS"). This small consortium was started by the State of Idaho in 2012, building atop the successful work that Idaho had already done to modernize its unemployment software infrastructure, with Iowa and Vermont also participating. (Iowa later dropped out and was replaced with North Dakota.) The project continued clear through 2019, with Idaho performing the software development work. At the beginning of 2020, Vermont raised the alarm, complaining of governance problems: specifically, Idaho was willing to let other states borrow iUS, but was unwilling to let them make any modifications to it, and naturally prioritized the needs of Idaho over those of Vermont or North Dakota. The governors of the three states tried to resolve these conflicts and, unable to do so, agreed to dissolve the iUS consortium. (This story was recounted by Vermont's Agency of Digital Services' Secretary John Quinn, in an April 2020 letter to the Vermont Daily Chronicle.)

IN PRACTICE: WyCAN

WyCAN was a multi-state unemployment insurance software consortium that included Wyoming, Colorado, Arizona, and North Dakota. The effort began in 2009 with a $62 million grant from the U.S. Department of Labor, in addition to funding from the member states. They teamed up via a cooperative purchasing governance agreement to build a monolithic system that would serve all of their needs. The states' benefits processes proved too different to be reconciled under a single system, and the work was abandoned, the unspent $47 million returned to the Department of Labor.

https://softwarecollaborative.org/cooperatives/wycan

How to Start an Intergovernmental Software Cooperative

Cooperatives follow many paths, and there is no single path to success. But there are some patterns among successful cooperatives. Anybody starting a new cooperative would do well to hew to the following prerequisites and practices.

IN PRACTICE: Intelligent Roadway Information System

IRIS is client/server software used by four states' transportation agencies to monitor and manage roadway traffic. It was created in 1999 by Doug Lau, an employee of the Minnesota Department of Transportation, who still maintains it today. IRIS is open source, and all collaboration between states is done in the traditional manner of an open source project. IRIS has no formal governance process, no interstate agreements, no additional support, and no project budget—there is only Doug.

https://softwarecollaborative.org/cooperatives/mn-iris

IDENTIFY SHARED NEED

Software cooperatives start with two or more government agencies that have the same need at the same time. Maybe it's a new federal mandate for states, maybe it's a new state program requiring localities to adapt, maybe it's a few states coincidentally implementing similar programs at the same time. Whatever the impetus, a simultaneous shared problem is the basis for any cooperative, and work has to begin with a clear, collective understanding of what that problem is.

START SMALL

It's important that co-ops start small; not 20 members, but 2. The biggest challenges of cooperatives co-vary with scale. More members means more problems. By starting small, those problems can be dealt with at a small scale, and new problems can be dealt with on a per-member basis as the co-op grows.

BUILD SMALL

It's also important that co-ops start by solving a small problem. They shouldn't start by building an entire unemployment insurance claims system. They should start by building a common application form, a common fraud-detection interface, or a shared platform for submission of eligibility documentation. Co-ops should create something valuable that can be implemented rapidly, so that members can learn how to work in this way.

ESTABLISH GOVERNANCE

The success of a cooperative hinges on its governance. Every member needs to understand what they are obliged to provide and what they can expect to receive. It is true that some types of cooperatives get by without such a document, but anybody looking to deliberately establish an intergovernmental software cooperative should formalize governance in a document that all members agree to. A governance document should specify member responsibilities, the process for agreeing what functionality will be produced, how expenses will be funded, and by what legal mechanism the software will be shared between members. The needs of members will sometimes be in tension, and the process of resolving that should be established at the outset. This process is simplified when an existing intergovernmental organization is expanding to include software sharing, because they'll already have much of this structure in place. The governance documentation for ActivitySim is a good example of the ideals described here.

ARCHITECT FOR GOVERNANCE AND NEEDS

The architecture of a cooperative's shared technical solution should reflect both the governance structure of the organization and the needs of its members. Are you producing ready-to-use software that every member can install on their own systems? Are you producing "roughed-in" software that every member will need to complete to integrate into their existing systems? Are you producing software as a service (SaaS) that the organization will house for its members' collective use? Each of these three architectures has benefits and drawbacks, but one of them is likely to best serve the intersection of the need to be addressed and the capabilities of the members.

IN PRACTICE: AASHTOWare

The American Association of State Highway and Transportation Officials (AASHTO) provides a suite of 16 transportation-management software packages for the benefit of its members, which include the transportation agencies of all 50 states, Washington D.C., and Puerto Rico. AASHTO is a 501(c)(3) that dates to 1914, and AASHTOWare dates to 1985. They outsource all software development, and their code is all closed source. States pay a licensing fee for access to AASHTOWare, with a site license for the whole suite running north of half a million dollars. This gives the project a substantial annual budget, within AASHTO's already-substantial budget.

https://softwarecollaborative.org/cooperatives/aashtoware

The question of the completeness of the software is important. Sometimes the needs of the members vary in important ways that make it impossible for them to use identical software. For example, every type of public benefits system is implemented in basically the same way, but they have significantly variant eligibility requirements and benefits, can require very different integrations with other states systems, and often use different terminology to describe the same things. When this is the case, SaaS isn't likely to be viable, and instead the co-op will need to create software that isn't quite complete, leaving each member to perform the finish work that will allow the software to meet their needs. This requires careful decision making when building that software. At every step of the way, the development team needs to work with that outcome in mind, constructing a modular system that will allow members to plug in the additional functionality that they require, instead of having to modify core functionality that will conflict with future updates.

INSOURCE OR OUTSOURCE

The co-op can either build custom software itself or can pay a software development vendor to build it. If the cooperative has funding, whether from members or a grant, the stability and duration of that funding might help to dictate the approach—a one-off grant might point toward outsourcing initial development work, while a sustainable, predictable funding stream might point toward hiring a persistent team of developers. If the cooperative's members have experienced software developers in their ranks, that might point toward building.

For a software development project that will span many years, it will likely be cheaper to employ a development team directly rather than contracting with a third-party vendor.

Note that it is not common for government agencies to directly employ experienced software developers. Before taking that approach, it's important to get an assessment of how competent the available developers are, and of their ability to work together as a cross-agency team.

IF PROCURING, USE AGILE CONTRACTING

If you've decided to procure custom software development services, follow the guidance found in the U.S. General Service Administration's "**De-risking government technology**," particularly around the **Agile contract format** and requiring **regular demonstrations of functioning software**.

You should not require the vendor team to work on-site. Developers do not want to do that, and you will wind up with a development team of people too junior to get out of a bad assignment. Absent organizational or political pressures to the contrary, it doesn't even matter where in the country the developers are. Otherwise-identical software developers in Washington, California, New York, Virginia, and Maryland command twice the salary of software developers in the Midwest or northern plains states, which means you can get twice as much for your money by being indifferent to developers' physical locations. This became common in 2020, and will likely remain normal post-Covid.

Finally, your cooperative must have two key employees or members assigned to the project: a product owner and an experienced software developer. The product owner works with users, stakeholders, technologists, and the vendor to envision the direction for the product, with an eye toward delivering value to end users as quickly as possible—they are the fulcrum on which the project's success hinges. And an experienced software developer is necessary to assess potential vendors, and essential for regularly reviewing the vendor's work output, specifically assessing the code, etc., for adherence to the requirements laid out in the contract.

USE MODERN SOFTWARE DEVELOPMENT PRACTICES

Whether you're building your own software in-house or hiring a vendor, it is crucial to use modern software development practices, and not the dated methodologies that tend to languish in government. These practices are well documented in the "Basic principles of modern software design" portion of the GSA government technology guide.

In short, rely on Agile software development and the associated practices of user-centered design, product ownership, DevOps, and building with loosely coupled parts—that is, identify user needs, address them to those users' satisfaction, and repeat. Get the resulting software in use by actual users as soon as possible, and incrementally deliver improvements to those users, ideally every two weeks.

IN PRACTICE: WinGAP CAMA

WinGAP Computer Assisted Mass Appraisal is a mass-appraisal tool used by county-level taxation authorities throughout Georgia. The collaborative effort began in 1987, with the first release of the then-DOS-based software coming two years later. A purpose-created non-profit organization houses and maintains the software, using $1,500/year membership dues from each of the 145 participating Georgia counties. This cooperative has thrived for decades, staying under the radar and operating on a shoestring budget.

https://softwarecollaborative.org/cooperatives/wingap

WORK IN THE OPEN

Successful cooperatives are more likely to work in the open than other government software projects. Their governance structure is public, meeting minutes are public, meeting agendas are public, their software is open source, their bug tracker is public, and their roadmap is public. GSA has documented the many benefits of working in the open, and all of that advice applies here. But an additional benefit is that working in the open makes it easier to attract new members to a cooperative. It is likely that a cooperative's competition is in the form of commercial software vendors, who keep busy responding to agencies' RFIs and RFPs, making sales calls, and setting up booths at conferences. It is unlikely that your cooperative will do any of these things, which might make it difficult to attract new members. Working in the open can compensate for this, providing your organization with a large footprint and making it easy for potential new members to learn about you and evaluate your offerings.

Conclusion

Software cooperatives have a long track record of reliably building and maintaining essential technical infrastructure for government. These non-conflicted organizations produce low-cost, high-quality software that solve pressing, specialized needs of agencies at all levels of government.

When agencies need new technical functionality, they should investigate cooperatively developed software prior to beginning any budgeting or acquisition activities. If no suitable software exists, agencies should seek to form new cooperatives with partners from similar agencies, to lower the individual cost of procurement and share the future burden of support and maintenance.

Government grant makers and appropriators should consider incentivizing funded agencies to form or participate in cooperatives, instead of awarding funding to states, for example, to build or procure the same thing 50 times over. A major custom software procurement has a low chance of success, while a reimplementation of something successfully implemented in a dozen states has a much higher chance of success, and a much lower cost.

Private-sector grant makers should consider funding non-profit software cooperatives to improve government service delivery. These organizations generally think of themselves as creatures of government, and are unlikely to consider applying for grants, so funders would do well to survey this landscape and reach out to co-ops that they are interested in supporting, to promote wider adoption of this lower-cost, lower-risk model of delivering public services.

Intergovernmental software cooperatives have quietly thrived for over half a century. Their impact has been substantial, but their work is poised to have quite a bit more impact over the next few years. To improve government service delivery, agencies and funders need to participate in and facilitate their rapid growth. The U.S. government has outgrown its legacy approach to technology, and we would do well to rapidly shift to this cooperative model.

Appendix A: Cooperative Sharing Models

COLLABORATIVE AGENCY DEVELOPMENT

Multiple agencies work together (either informally or with a memorandum of understanding) on software development from the start, with each agency contributing in the form of staff or contractors' time. The resulting software may be reused as software as a service (SaaS), or as executables, or as source code distributed to participating agencies.

Example: ReEmployUSA

COLLABORATIVE ORGANIZATIONAL DEVELOPMENT

Multiple agencies are members of an organization, and that organization performs software development work on behalf of the member agencies, via direct hires or contracting. Member agencies may contribute financially, or work may be supported by external funding (e.g., from a higher level of government or from a grant). The organization may have been created expressly for the purpose of software collaboration, or it may be an existing interagency organization. The resulting software may be reused as SaaS, as executables, or as source code distributed to member agencies.

Examples: Digital Towpath, Association of Oregon Counties' Integrated Road Information System

BUILT HERE, OTHERS USE

An agency builds software and releases the software or source code publicly, and other agencies then use that software. The software continues to be housed by the original agency.

Example: Notify

BUILT HERE, OTHERS CONTRIBUTE

An agency builds software and releases the source code publicly, and employees of other agencies contribute their modifications. It may continue to be housed by the original agency, or it may transition out to be community-supported.

Example: OpenTripPlanner

BUILT EXTERNALLY, AGENCIES CONTRIBUTE

A non-government organization has created open source software, and it becomes used within government agencies, which then make contributions to the software so that it can better serve their needs.

Example: QGIS

TOP-DOWN

A "parent" government builds software and shares it with "child" governments (e.g., a state provides it to counties) as SaaS or as executables/source code.

Examples: **HURREVAC**, **WinGAP Computer Assisted Mass Appraisal**

BUILT OPEN, THEN PRIVATIZED

An agency builds software that is in the public domain, then a private organization takes it over and redistributes it to other agencies for purchase or reuse. This doesn't necessarily mean that it is commercialized, but it may be.

Examples: **SQLite**, **Open Path**

BUILT COMMERCIALIZED

An agency hires a vendor to build custom software for them, and the vendor retains ownership. The vendor then resells the software to other agencies, who may not be aware that the software was originally built for another agency. This isn't strictly "cooperative" or "sharing," but it does provide some of the same benefits as the other models, and is included here for completeness.

Examples: **Deloitte's HealthInteractive**

Appendix B: Inventory of Cooperatively Developed Software Projects

This list includes every cooperative software project that we are aware of. If you are aware of other projects we should include, please contact us.

- AASHTOWare: This suite of highway construction and maintenance tools is produced by the American Association of State Highway and Transportation Officials, and is used nationwide.
- APHL Informatics Messaging Services: The Association of Public Health Laboratories provides this cloud-based SaaS for the hosting and exchange of health data, for exchange between all U.S. states, the federal agencies, and hospitals.
- ActivitySim: This organization produces open source travel behavior modeling software, and is comprised of regional transportation planning agencies in six states.
- American Association of Motor Vehicle Administrators: This organization, which claims every state as a member, provides 18 different shared technology services for state motor vehicle agencies, including non-premises software and SaaS.
- BizPal: A free online service that helps Canadian businesses identify the permits and licenses they need, and how they can obtain them, managed by a partnership involving hundreds of governments at the federal, provincial, territorial, and municipal levels.
- CONSUL: Created by Madrid municipal employees, this open source citizen participation tool is in use by governments in 35 countries.
- CPE Audit Service: The National Association of State Boards of Accountancy provides this SaaS tool for states to conduct audits of licensee compliance. It is used by eleven states and territories.
- Census and Survey Processing System: This Windows-based public domain software package is used for entering, editing, tabulating, and disseminating census and survey data. Created by the U.S. Census Bureau and a vendor, it's used in over 160 countries.
- Code for Development: The Inter-American Development Bank—a membership-based financer of Latin American and Caribbean economic development—has created 32 open source software packages for their members' benefit, ranging from urban growth prediction to web forms generation.
- Concierge: Developed by Canada and The Netherlands, this OAuth2 and OpenID microservice is used for handling user registration, login, and SAML2 single sign-on.
- Digital Towpath: Nine New York municipalities have teamed up to create this content management system to host web sites, increase communication with residents, and manage electronic records.
- Drupal WXT: An open source Drupal distribution created by the Government of Canada to facilitate compliance with the country's language and accessibility requirements, used by governments around Canada.
- Electronic Verification of Vital Events: The National Association of Public Health Statistics and Information Systems verifies identities by matching queries against United States birth certificate databases, via a network which nearly all U.S. states and territories participate in.

- Evergreen: An open-source integrated library system, originally created by Georgia, but now housed by a membership organization, which maintains it for the benefit of the thousands of libraries that use it, including in many U.S. states.
- GovCMS: This Drupal-based content management system is created and maintained by Australia's national government, and it is available in SaaS, PaaS, and self-hosted options. It's in use across 96 organizations (agencies and ministries) at all levels of Australian government.
- HURREVAC: The National Hurricane Program produces this web-based tool for storm tracking and decision support, for the benefit of local emergency managers in hurricane-prone states.
- Intelligent Roadway Information System: This tool is used by transportation agencies to monitor and manage interstate and highway traffic. It's created by the Minnesota Department of Transportation, and shared informally with three other states.
- International Registration Plan: This SaaS is operated by a national member organization of the same name, which exists to facilitate a reciprocity agreement to share revenue from commercial vehicle registration fees based on miles driven in each state. This data storage system allows every U.S. state to share vehicle data.
- Internet Unemployment System: This multi-state unemployment insurance software consortium was led by Idaho, and included three other states at various times. It wound up only being useful to Idaho, and is no longer a consortium.
- Known Traveller Digital Identity: A joint project of France, the Netherlands, and the World Economic Forum, this early-stage project seeks to verify travelers' identities when traveling between member countries.
- Library Simplified: This collection of middleware, server software, collections management tools, and mobile client applications is used by libraries to deliver e-books and audiobooks to their patrons. Created by the New York Public Library with a federal grant, it's now in use by libraries across the U.S.
- LocalGov Drupal Club: Several towns in England teamed up to collaboratively build a collection of open source modifications to Drupal 8 to address needs common among town councils.
- Malware Information Sharing Platform: This Luxembourg-based project is an open source platform for collecting and sharing cybersecurity indicators and threats. Several EU member states participate.
- Minnesota Educational Computing Consortium: An early cooperative, this was created by Minnesota school districts in the 1960s, which pooled their resources to purchase time on mainframes. They went on to produce games that were so popular that they were sold nationally, including Oregon Trail, Number Munchers, and Word Munchers.
- Multiphysics Object Oriented Simulation Environment: This open source finite-element, multiphysics framework software package was created by the Idaho National Laboratory, but is used and contributed to by other government agencies and by vendors.
- NAIC: The National Association of Insurance Commissioners provides a suite of software tools for state insurance regulators in every U.S. state, such as a life insurance policy locator and a series of API endpoints to support regulators' needs.

- NASWA: Three different shared services are provided to states by the National Association of State Workforce Agencies, including transmitting unemployment insurance claims between agencies and employers, coordinating wage data between states, and preventing unemployment insurance fraud.
- New York Community Officials Data Exchange: This data-sharing platform was created by three New York localities teaming up to deal with blight.
- New York Real Property System: The New York State Office of Real Property Tax Services created and maintains this Windows-based Computer Assisted Mass Appraisal software for the benefit of localities in New York.
- Nlets: This organization counts every state law enforcement agency as a member, and uses their data-sharing platform to support queries about individuals, drivers license records, and Interpol records.
- Notify: An open-source SaaS tool, this is used by a host organization to allow members to send text messages, emails, and postal letters with a simple API call. It was created by the UK's Government Digital Service, and has been reimplemented by the Canadian, Australian, and United States governments.
- Open Path: This open-source suite of tools is used by governments to provide services to people experiencing homelessness, gathering both client-level data and data about housing and services. It was created by Boston, and is now used in several states.
- OpenFisca: An open source platform to write rules as code, allowing economists and public administrators to simulate the economic impact of changes to taxation or benefits. A project of the French government's digital service agency, Etalab, this decade-old project is in use in several other countries.
- OpenTripPlanner: This suite of open source software is used by transit agencies to create travel itineraries that span transportation types, allowing people to plan trips on the transit agency's website. It was created by the transit agency of Portland, Oregon, and is now in use in a half-dozen U.S. states.
- Oregon's Integrated Road Information System: The Association of Oregon Counties, a membership organization, provides this cost accounting software used in the maintenance and operations of county road departments. The decades-old, Windows-based software project is in use by most Oregon counties.
- QGIS: This popular open source GIS tool wasn't created by a government agency, and it's not maintained by one, but it routinely adds features that were developed by or funded by local governments that required that functionality.
- ReEmployUSA: Five states have teamed up to form this unemployment insurance software consortium, built by a vendor, with each state hosting their own copy of the software.
- SILVAH: The U.S. Forest Service provides this Windows-based desktop software for making silvicultural decisions in hardwood stands of the mid-Atlantic and upper Appalachian region.
- SQLite: This ubiquitous database software, running on every computer and handheld computing device, was created by a U.S. Navy contractor in 2000. It is in the public domain, which allows it to be incorporated into vast numbers of software products. All government agencies use it, but few are likely to have any idea that they do so.

- **Sedipualba @**: An electronic management service for administrative activities of governments, this Spanish-language SaaS tool is provided under a cost-sharing model.
- **Southeast Consortium Unemployment Benefits Integration**: This two-state unemployment insurance software consortium was funded by a U.S. Department of Labor grant, and is in use in North and South Carolina. It's cloud-based and built by a vendor.
- **State and Territorial Exchange of Vital Events (STEVE)**: Fifty-three states and territories participate in this vendor-built system to exchange birth and death records. It's a product of the National Association of Public Health Statistics and Information Systems.
- **Utah Courts' Online Dispute Resolution**: Utah is piloting the use of online dispute resolution for small claims cases at select courts throughout the state.
- **Veterans Health Information Systems and Technology Architecture**: The Departments of Veterans Affairs created this Windows-based client/server health administration system for use in its 1,700 hospitals and clinics. Released into the public domain, the WorldVistA organization was created to support VistA's use around the world, where it's used by both public- and private-sector healthcare facilities.
- **VisRate**: Washington State's County Road Administration Board created this Windows-based tool for county transportation departments to collect pavement distress data, used to monitor road health.
- **WinGAP CAMA**: This Windows-based client/server mass-appraisal tool is created and maintained by employees of a membership organization comprised of Georgia counties' property tax appraisal departments. It's used to track privately owned assets (buildings, vehicles, boats) to track their appraised values for the purpose of property taxes.
- **wgrib2**: The National Weather Service's Climate Prediction Center created this open source utility to read and write weather data, with representatives from other government agencies (including NASA and the Netherlands Institute for Radio Astronomy) contributing in the form of improvements to the software.
- **WyCAN**: This four-state unemployment software consortium started in 2009 with a $62 million grant from the U.S. Department of Labor. They tried to build a monolithic system that would serve all of their needs, but the states' different unemployment benefits processes made that impossible, so the project was terminated.
- **X-Road**: The Estonian government had a vendor build this secure data exchange system, and then signed an MOU with Finland to collaborate on further development. The open source system is now in the hands of a membership organization created to house the software, and counts two more countries among its users.
- **Zephyr Foundation**: This membership organization provides open source travel analysis software for state and municipal transportation agencies in four states. They produce several different software packages to that end.