Federal Source Code Study (FSCS)

by

Joseph Castle, PhD

Code.gov

Executive Summary

Publishing software and its associated source code for public use is a new phenomenon for U.S. federal government agencies. In August 2016, the White House issued the Federal Source Code Policy: Achieving Efficiency, Transparency, and Innovation through Reusable and Open Source Software (FSCP). The FSCP mandated executive-level agencies to publish at least 20% of their custom developed code as open source software (OSS). OSS is software that can be shared within a community of developers through accompanying licenses hosted in online code sharing platforms. The federal government has the responsibility to account for public spending, including spending for IT. The publication of OSS is one way the public can know about government spending. OSS additionally benefits the public by providing access to code, thus, making it the "People's Code."

From 2016 to 2019, the progress of executive branch agencies in implementing the FSCP was mixed. This study examines whether and how organizational factors – cultural beliefs, public engagement, structural dimensions, and organizational location – affect agency policy implementation. The study uses the publication of OSS as an indicator of effective policy implementation, and it identifies the factors that hinder or aid publishing OSS.

To arrive at a general understanding of agency efforts at policy implementation, I collected data from GitHub's application programming interface (API) and created a list of 23 of 24 executive-level agencies that published OSS both before and after the FSCP was issued. From these agencies, 25 participants from 20 agencies agreed to participate in the study. These participants were from software development units that minimally, moderately, or frequently published OSS. The sample consisted of participants from units mostly located outside a Chief Information Officer (CIO) office that focused on software development and data science

activities. Grounded theory provided an approach for data collection with interviews and document collection, leading to continuous analysis for generating a theory of policy implementation for OSS publication. Units more frequently published OSS when they expressed views complementary to those of their parent organization and held advantageous cultural beliefs; practiced more and more varied public engagement through two-way communication, events, and electronic tools; had structures with less centralization, more formalization, more differentiation, and more coordination; and were located in the "middle" of an organization with fewer hierarchical layers. Additionally, some units expressed both cautionary and advantageous cultural beliefs suggesting beliefs alone are not enough to allow units to publish OSS.

This study contributes to policy, public administration, and organization theory literatures. It enhances scholarship by examining a new phenomenon and aids practitioners by providing implications for consideration when implementing policy.

Table of Contents

List of Appendices

List of Figures

List of Tables

Chapter 1 Introduction

Computer software is essential for the daily operations of industry and government. "More and more major businesses and industries are being run on software – from movies to agriculture to national defense" (Andreessen, 2011, p. 1). Software drives the world's largest commerce platforms (e.g., Amazon, Wal-Mart, GSA's Integrated Acquisition Environment), music delivery companies (e.g., iTunes, Spotify), and video entertainment (e.g., Netflix). Further, software supports national defense through intelligence, communications, logistics, and weapons guidance.

Software is made of a collection of human-readable source code[1] files that a computer executes (see Figure 1.0). The U.S. federal government has over 6,800 software projects listed on the Code.gov website with 80% being open source software (OSS) ("Sharing America's Code," n.d.). These software projects are reused at varying rates, resulting in software development efficiencies and cost savings to the user. Examples of reused federal OSS projects and associated impacts include:

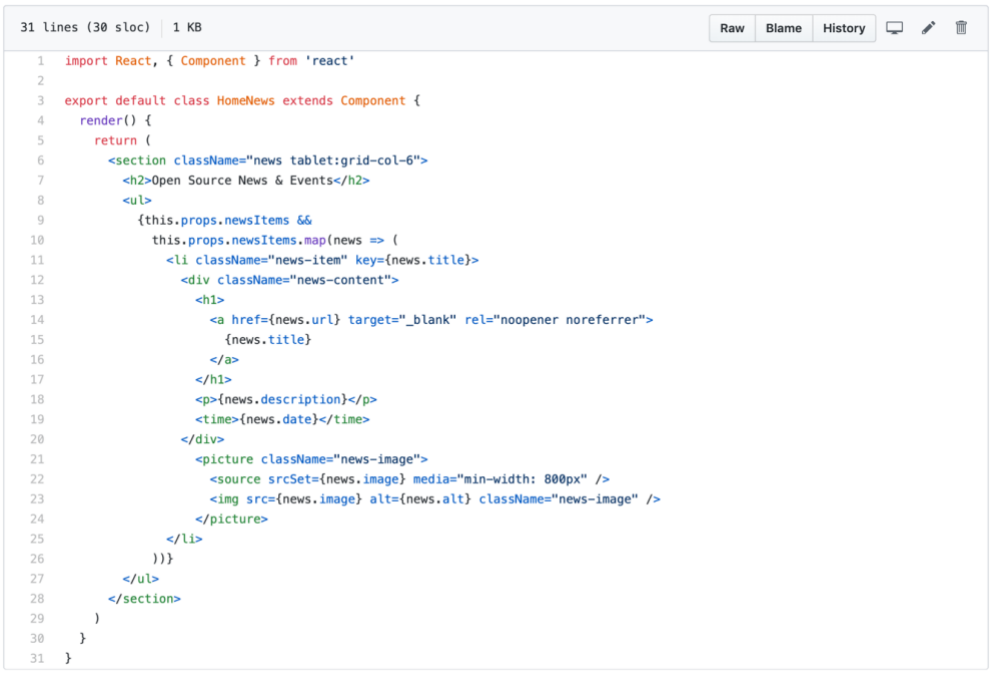
- The U.S. General Services Administration's (GSA) U.S. Web Design System (USWDS) creates a better web and mobile experience for end users. It provides templated code for development of modern user interfaces, responsive capabilities, and 508 compliance[2] ("United States Web Design System," 2020). Over 150 federal, state, and local governments are "saving around $100,000 per project on design and developer resources when using USWDS" (Strenio & Ashida, 2019, p. 1).

[1] Software can either be closed source software (CSS), where the source code is not available to users, or open source software (OSS). OSS can "be freely used, changed, and shared (in modified or unmodified form) by anyone" ("The Open Source Definition (Annotated)," n.d., p. 1).
[2] The Rehabilitation Act of 1998 included Section 508: Federal Electronic and Information Technology. This section of the law requires federal agencies to make technology (i.e., computers, software, electronic office equipment) accessible to individuals with disabilities.

- The Department of Energy's (DOE) Lawrence Livermore National Laboratory (LLNL) Zettabyte File System (ZFS) for Linux is a file system for supercomputers ("LLNL Software Portal," 2020). The federal government uses ZFS for Linux on the Sequoia supercomputer to conduct numerical simulations of nuclear weapons performance. Industry also uses ZFS for Linux. A notable example of industry use is Netflix, Inc., which uses ZFS to store and stream media. It is one of the most popular LLNL software projects with over one thousand copies (i.e., forks[3]) and over five thousand stars (i.e., likes) on GitHub.com by users.
- The Environmental Protection Agency's (EPA) Water Network Tool for Resilience (WNTR) is an application used to simulate and analyze water distribution networks under disaster scenarios ("Water Network Tool for Resilience (WNTR) User Manual," 2019). Drinking water systems are a U.S. national priority because they are one of 16 critical infrastructures. The simulation software generates network models, simulates water quality, identifies repair strategies, and visualizes results. Many hackathons,[4] hosted by federal agencies, use this application to teach individuals to code and interact with the federal government. Users employ the application to run their own water disaster simulations and remediation scenarios. U.S. and international governments use the application.
- The National Security Agency's (NSA) WALKOFF is a security orchestration automation and response (SOAR) framework ("WALKOFF," n.d.). WALKOFF supports many tasks including cybersecurity, system administration, analytics, and

3 Forks are a copy of the original source code. When users fork the source code, they modify the code without affecting the original code. After modification, users can submit changes to the owner of the original source code.
4 Hackathons are events where computer programmers from inside and outside government collaborate on software development projects.

internet-of-things. "It currently has 25 open source apps with which it will work, ranging from Linux scripting to Watson facial recognition to Tesla to Weather" ("10 Open Source Projects to Try Over the Holidays," 2017). U.S. and international governments use WALKOFF and associated applications for a multitude of projects.



31 lines (30 sloc) | 1 KB — Raw | Blame | History

```
import React, { Component } from 'react'

export default class HomeNews extends Component {
  render() {
    return (
      <section className="news tablet:grid-col-6">
        <h2>Open Source News & Events</h2>
        <ul>
          {this.props.newsItems &&
            this.props.newsItems.map(news => (
              <li className="news-item" key={news.title}>
                <div className="news-content">
                  <h1>
                    <a href={news.url} target="_blank" rel="noopener noreferrer">
                      {news.title}
                    </a>
                  </h1>
                  <p>{news.description}</p>
                  <time>{news.date}</time>
                </div>
                  <picture className="news-image">
                    <source srcSet={news.image} media="min-width: 800px" />
                    <img src={news.image} alt={news.alt} className="news-image" />
                  </picture>
              </li>
            ))}
        </ul>
      </section>
    )
  }
}
```

Figure 1.0. Source Code File.

The federal government spends approximately six billion dollars on software each year (Johnson & Sears, 2017), representing 7% of the information technology (IT) budget of the Chief Financial Officers (CFO) Act agencies ("Government-wide Information Technology Agency Summary," 2017). This purchasing of software frequently results in duplicative and wasteful spending annually because the software is not publicly shared. Agencies purchase closed source software (CSS) and services to customize as needed, leading to software purchasing and support that is costly, allows for little to no reuse, results in duplication of effort and spending across

agencies, and overall yields poor quality (Mann, 2002; Rosenberg, 2007). Moreover, publicly funded software products are not open to the public even though the software is purchased with taxpayer dollars. The federal government has the responsibility to account for public spending, including IT spending. The publication of OSS is one way that the public can know about government spending, and OSS additionally benefits the public by providing access to code, thus, making it the "People's Code." Because software is a public good, U.S. federal organizations should be making their software open to the public, but federal organizations are lacking in publication of OSS.

Prior to the release of the Federal Source Code Policy (FSCP) in August 2016, only three of the 24 CFO Act agencies[5] published OSS. Nearly two years later, in November 2018, 21 of the 24 CFO Act agencies published OSS with varying frequency[6], as determined by data mining and analysis of the GitHub application programming interface (API) using Irizarry's (2018) python scripts. The data showed that even though most CFO Act agencies are publishing OSS, the amount decreased after the policy was released, which is counter to intended outcomes of the FSCP. Furthermore, the overall amount of OSS these agencies published is relatively small, especially considering the amount spent on software and the overall size of the federal IT budget.

Purpose of the Study

The purpose of this study was to examine the degree of agency implementation of the FSCP, using OSS publication as an indicator of implementation and of compliance with the policy. The research examined the federal government OSS landscape in 2019 and early 2020 as the government pivoted from being a purchaser and creator of CSS and a consumer of OSS to becoming a publisher of OSS for public consumption. Guiding this research in its exploration of

[5] Located in Appendix A: Chief Financial Officers (CFO) Act agencies.
[6] Located in Appendix B: GitHub.com Metadata of Open Source Software (OSS) Published by CFO Act Agency.

FSCP policy implementation was the question: Why do some federal agencies publish OSS and others do not?

Theoretical Framework

Using Hatch's (2018) overview of organizational theories as a starting point, I explored the five pillars of organizations: environment, technology, culture, social structure, and physical structure. From this initial exploration, I honed in on variations of those variables that I observed from my engagement with CFO Act agencies that aided or hindered OSS publication. Presumably, units in all CFO Act agencies adhere to the same software development practices, policies, and security standards as prescribed by the Office of Management and Budget (OMB) through department-level policies. Even with an executive order and OMB oversight, agencies moved slowly to publish OSS; in some cases, they published less after the mandate, signaling larger systemic difficulties with technological implementation. Arguably, these difficulties stem from larger organizational dynamics. Through my continued interactions with CFO Act agencies, it became clear that some of these variables (e.g., culture and social structure) had more of an impact than others on the ability of agencies to publish OSS. By exploring the history of OSS, reviewing organization theory scholarship, and refining the variables, it became evident that cultural beliefs, public engagement, structural dimensions, and organizational location were key.

Cultural beliefs. Beliefs are a key component of organizational culture and can support organizational performance. According to Schein (2010), beliefs supported organizations in making predictions and selecting actions, and these beliefs played an important role in understanding what was observed and enacted in particular situations. Cultural beliefs could explain whether units implement policy.

Public engagement. Public engagement is a component of a broader public participation construct, along with communication and consultation. Descriptive in nature, public engagement centers on the action government units take to involve those outside their unit. As Feldman and Khademian (2007) found, organizations that employed a multitude of engagement formats ultimately led to more effective programs because public engagement enabled individuals to come together within a technical domain to ask questions and solve particular problems. Public engagement may well complement the community aspect of building and publishing OSS.

Structural dimensions. Social structure in organizations provides arrangements for gathering information to achieve a collective outcome, the very reason for organizational life. As Hatch (1997) noted, the social structure of organizations consisted of "relationships among social elements including people, positions, and the organizational units to which they belong (e.g., departments, divisions)" (p. 161). Researchers in the field of technology-structure agreed that centralization, formalization, differentiation, and coordination were the major theoretical dimensions of structure, and each dimension, when isolated, affected organizations' responses to environmental disturbance (Fry, 1982; Scott & Davis, 2007; Woodward, 1981). This finding suggests that structural dimensions could influence the frequency of units' OSS publication as the executive order can be viewed as an environmental disturbance.

Organizational location. An additional structural dimension not included in the technology-structure literature is organizational location. Organizational location should not be confused with physical location. It instead refers to where a unit resides hierarchically (i.e., its location on an organizational chart). The organization design literature suggested that a unit's location should be selected to achieve internal consistency in order to match the organization's response to the environment (Harris & Raviv, 2014; Mintzberg, 1983). Organizational location

is a necessary variable to explore policy implementation, as evidenced by units' OSS publication.

Research Approach

For this research, I utilized Creswell's (2009) common aspects of qualitative research that described how the researcher is situated in the study for data collection and analysis. As the researcher, I was the key instrument in the study. I employed a Straussian grounded theory approach that was grounded in data systematically gathered and analyzed. Data were collected from multiple sources including metadata, interviews, and artifacts. My epistemology shaped the analysis of these data. I have worked in federal government organizations for over 15 years; I believe that OSS is beneficial for government organizations and the public and that the federal government should release more code.

Significance of Study

U.S. federal government agencies have been consuming OSS for the past 70 years but are new to publishing their own source code. Agency OSS publication is a unique policy output and a tangible indicator of effective policy implementation. The analysis adds to the limited literature on the intersection of the federal government and OSS publication, and it applies organizational theoretical concepts to a new phenomenon.

Chapter Organization

Chapter 2 begins with the background and evolution of OSS and the U.S. government's involvement with and contribution to OSS. It traces how software complemented hardware development since the 1940s. It also provides insight into how OSS often is considered a movement and a process.

Chapter 3 introduces the grounding for the conceptual framework used in this study. The framework is based on research related to the organizational variables affecting policy implementation for OSS publication, specifically examining whether and how such factors hinder or aid federal agencies in implementing new technology. Chapter 4 describes the research design and methodology used to investigate the research questions. The chapter begins with a description of the approach to research and presents the data collection and analysis processes.

Chapter 5 presents the findings. Finally, Chapter 6 concludes by briefly summarizing the findings and suggesting ways of moving toward a theory of policy implementation for publishing OSS. I also discuss the implications and limitations of the study and offer recommendations for future research.

Chapter 2 Background and Evolution of OSS

OSS consists of computer source code incorporated into an application or software that can be copied, modified, and shared within a community of developers through accompanying licenses hosted in online code sharing platforms. The history of OSS is intertwined with the broader history of hardware and software development (Ceruzzi, 2012; Isaacson, 2014; Levy, 2010; Roberts, Hann, & Slaughter, 2006). Computing innovations occurred with some overlap, with significant events leading to the development of OSS as a concept, movement, and process. This chapter presents relevant research pertaining to several domains: public administration IT, the role of chief information officers (CIOs) in software, OSS research, OSS as a movement and process, and the federal government's intersection with OSS.

Research on Public Administration IT

Scholarship on public administration IT (PAIT) developed from the need to have a specific research literature pertaining to the management of IT government, as other disciplines and fields (e.g., business management, information technology, information science) discussed IT management. This need was in part due to the distinctive mission of public agencies to provide public service. As Holden (2003) stated, "federal agencies rely extensively on information technology (IT). Arguably, public administration should be driving the theory, policy, and practice for managing these increasingly important resources" (p. 53). PAIT literature emphasized a centralized, top-down view of managing IT in the U.S. federal government before and after formalization of a Chief Information Officer (CIO) position. This work focused on IT management from the 1970s to the early 1990s. The scholarship described

three phases: Management Information Systems (MIS), Information Resources Management (IRM), and Information Age (Holden, 2003).[7]

The first phase, MIS, spanned the 1960s through the mid-1970s, when management narrowly focused on IT within data centers with little consideration given to management of IT in other parts of the organization. IT professionals often worked in backroom data centers with little management oversight and were charged with facilitating clerical efficiencies for the organization. Besides the IT professionals, users interacted with data processing technology, often on mainframes, through batch processing with punch cards for computations and generating print-out reports. Most computer makers and employees in large companies generally thought that these computer systems would never meet managers' expectations or needs in terms of utilizing IT to accomplish discrete tasks in short amounts of time (Dearden, 1972).

The Paperwork Reduction Act (PRA) (1980) ushered in Information Resource Management (IRM) as a replacement for MIS from the mid-1970s through the mid-1980s. The PRA called for federal agencies to manage IT as an organizational task similar to other organizational functions (e.g., acquisition, human resources) with emphasis on both information (i.e., collection of meaningful data) and technology (i.e., tools performing activities for processing inputs to outputs). Technology migrated from purely mainframe computers hosting data centers to microcomputers for daily use by many in the workplace. This required agencies to develop "architectures" to support this new availability of IT, which was a "departure from the earlier philosophy of managing IT, because [IT] systems [now] existed outside the physical and management control of a central data center" (Holden, 2003, p. 59). Managers acquired computing resources as they saw the relevance of technology for business use and had to manage

[7] An extensive review of the three phases of PAIT can be found in Holden (2003).

people and technology for a common output (Caudle, 1987). The organizational environment changed from a centralized, command and control environment to an environment that allowed employees more autonomy as they determined ways to be more effective with computers and technological tools (see below).

The Information Age began in the mid-1980s and continued through the early part of the 1990s and "described a future state where information [would] be quickly and universally accepted in electronic form" (Holden, 2003, p. 64). Internet technologies and communication networks became essential to tie technological tools together for broad information sharing. This required transformation of organizations and of the personal use of information because organizational needs could only be achieved by broader organizational technology goals. For public organizations to be effective, managers and workers needed technological skills, and they needed to develop IT strategies and IT structures to support planning, fit, performance, and acquisition of IT (Applegate et al., 1999).

PAIT continued to change the management of IT: "relegating the management domain to 'computer people' [was] no longer sufficient" (Holden, 2003, p. 69). Information technology in organizations became as important as other pillars (e.g., accounting, manufacturing, human resources). Organizations had to consider ways to leverage IT to accomplish other primary functions. Organizations became more efficient with technology and utilized it to establish organizational goals and objectives. Information technology became something that every employee must know and understand.

PAIT research focused on managing IT from an organizational standpoint. It described the management of both information and technology in public organizations. Yet it discussed neither the formal role of the federal CIO nor specific technologies in-depth (e.g., software,

hardware, mobile devices). The subsequent section begins where PAIT ended in the late 1990s and discusses the intersection of the CIO and software purchasing and development.

Federal CIOs and Software

The Clinger-Cohen Act of 1996 established the CIO position in each of the 24 CFO Act agencies. The president appoints nine of these CIOs, and the department secretary or agency administrator appoints the remaining 15. All of the CIOs are executives in their respective agencies. CIOs are responsible for "managing IT in six areas: leadership and accountability, strategic planning, workforce, budgeting, investment management, and information security" ("CIO Management Responsibilities Remain a Challenge for Most Agencies," 2018, p. 1). Practically, CIOs are responsible for all aspects of IT, from managing servers for Internet connectivity and sending email, to purchasing and customizing software to fulfill basic functions (e.g., human resources and finance) and meet customer needs (customer relationship management, Software-as-a-Service), moving network and data centers to the cloud, administering hardware and mobile devices, and ensuring privacy and security compliance. CIOs also are responsible for implementing policies created by the U.S. CIO (USCIO) in OMB (e.g., Open Data Policy: Managing Information as an Asset (M-13-13)) and laws enacted by Congress (e.g., Digital Accountability and Transparency Act of 2014).

CIOs procure and customize software to meet agency needs. In many cases this is accomplished through acquiring technology products and contractors, whom government project and program managers supervise. Agencies often have multiple CIO offices responsible for software development; however, software development can occur outside the CIO office depending on the purpose and skills of the office (e.g., data science). As Scott (2017) explained, the majority of code in an agency, typically created in a CIO office, was custom developed for

old and new applications alike, from mainframe to the modern cloud. Developing for new and existing applications potentially can result in inefficient code maintenance in the longer term. This occurs within agencies when individuals follow the "shiny object syndrome." These individuals want to move to the latest software but they do not always consider the broader technology ecosystem. Moreover, each agency attests to needing to customize software because of its "unique" requirements even though open source software might well meet the agency needs.

Moving from the centralized function of IT management and the role of CIOs, this study offers a more targeted examination of the development of OSS and an exploration of the factors that influence the publication of OSS. Attention turns next to a historical overview of OSS.

Research on OSS

The modern computing era of the 1940s and 1950s consisted of large computational machines, funded by the U.S. government and hosted at academic institutions. Computers at this time were used to calculate artillery trajectory tables in support of World War II. These machines, built for war, were initially designed to complete a specific task such as solving equations or deciphering codes (Isaacson, 2014). During this time, computer programming consisted of flipping switches to change electrical signals or using paper punch cards for creating and executing lengthier calculations. Computers accepted programs by storing instructions, allowing for algorithms to execute complex computational logic and lengthier problem sets. Some of the earliest elements of OSS, community and code sharing, can be traced to Grace Hopper's work at the Harvard Computational Laboratory with her efforts in building a compiler for the Common Business Oriented Language (COBOL) (Beyer, 2009).

The mainframe and the microcomputer era of the 1960s and 1970s began with most hardware and software bundled to support mainframes. Realizing considerable commercial potential, computer companies came into existence with a focus on mainframe computing for large organizational computing purposes in government, industry, and academia. International Business Machines, Inc. (IBM) sold the majority of mainframes and bundled software, creating a monopoly. The U.S. government threatened lawsuits, leading IBM to unbundle software from the mainframe and allowing new opportunities for competition within the nascent software industry. Soon code matured into full-fledged programming languages, mimicking human language, including FORTRAN (i.e., Formula Translation) and COBOL.

Two important events during this time led to OSS emerging as a distinct phenomenon. First was computing in contrast to sequential batch processing, or time-sharing, that involved students at the Massachusetts Institute of Technology (MIT) who worked on mainframes at scheduled times. Each user's work literally was left in the open, on the computer, resulting in the sharing of programming innovations freely for others to benefit from, build upon, and share again. Building upon others' work through trial and error, or "hacking," laid the groundwork for a different approach to software development. These "hackers" used clever programming, resulting in innovation, differing development styles, and technical virtuosity. Additionally, these individuals adhered to a "hacker's ethic": "The precepts to this revolutionary hacker ethic were not so much agreed upon and discussed as silently agreed upon" (Levy, 2010, Chapter 2 ¶ 3). The ethos included:

- Access to computers,
- Belief that information should be free,
- Mistrust in authority,

- Judgment of hackers based on software development proficiency,
- Creation of art and beauty with computers, and
- Belief that computers can change one's life for the better.

Second was the development of the personal computer (PC). Large-scale computers, mainframes, and legacy computers from WWII were inaccessible to most individuals at work or at home. Size, cost, and computer engineers' general belief that individuals would not want or need a PC discouraged computer companies' development of the PC. However, during this time, a group of hobbyists, mostly children of computer engineers at corporate computer companies, were driven by idealistic visions of the 1960s counterculture to build PCs (Ceruzzi, 2012). These machines were important in supporting the hacker's ethos and provided freedom to the user to create and use computer programs for gaming, word processing, accounting with spreadsheets, and communication. IBM also built a microcomputer and created the term "personal computer" to separate its product from the competition, but it was the hobbyists and hackers who expanded the use and drove the new personal computing and associated software industry.

Personal computing became more accessible because of work by university hackers and hobbyist computer club participants. This occurred for many reasons: 1) computer parts were becoming smaller, affordable, and accessible. 2) These smaller computers were being glorified in electronic magazines, such as the Alltair featured on the cover of *Popular Electronics*. 3) Computer clubs (e.g., Homebrew Computer Club) comprised of individuals who often were the children of parents who worked in corporate computer companies (e.g. IBM, DEC), coming together and sharing knowledge to build personal computers (Isaacson, 2014).

The 1980s, also known as the PC era, continued with the growth of personal computer construction and software development moving from operating systems to applications for personal and office productivity. Computers and software were still limited mostly to businesses and a few hackers and hobbyists. The free sharing of software ended when companies began selling microcomputers with software already installed, with its source code not available to the user. As demand for products from the general public increased, these new personal computer companies flourished. For the hacker and hobbyist communities, a philosophical rift began when the CEO of Micro-soft (later Microsoft, Inc.), whose company built operating systems for the Altair 8800 and later MS-DOS for IBM PCs, wrote an open letter to hobbyists in *Popular Electronics*, saying software should no longer be free (Gates, 1976). For these new computer companies, competition to innovate software generated an increased need for software copyrights, licenses, and limited sharing. Furthermore, "sharing and collaborative development were beaten down by deregulation, privatization, and the enclosure of source code by proprietary players" (Weber, 2004, p. 52). For most of the 1980s and into the 1990s, the majority of software was closed source and proprietary, meaning hackers, hobbyists, and developers no longer had access to source code. Some non-corporate developers started OSS non-profit foundations and began open source projects, but corporations often considered this source code not mature enough for corporate use.

The Internet era of the 1990s consisted of three main approaches to software development: 1) software and hardware as tightly bundled; 2) software including the operating system as unbundled; and 3) free and open source software, allowing for software to be modified by any user (Isaacson, 2014). The resurgence of OSS occurred because of the Internet's communication capabilities and the development of web technologies. Developers could now

share software through email or by posting on a website. Developers created server and web software (e.g., Linux, Apache, Netscape), which resulted in product demand by computer professionals and general users alike. Instead of software being developed within a company, it was developed and posted online where any developer could improve it and post changes for review and acceptance by a community of developers. In a short period of time, Internet and web technology software became more widely distributed. General users, corporations, and governments then began to consume the software for daily operations. A new norm in computing evolved: organizations regularly used the Internet, web, and open source models for corporate operations (Isaacson, 2014).

OSS as a movement. OSS was part of a broader philosophically constructed computing movement, which was based on the belief that software should be created, shared, and improved upon communally in the public domain (Constantine, 2007; Isaacson, 2014). The movement began as corporations pursued a profit motive of selling software by increasingly selling restricted software. These corporations often ridiculed hackers and hacker products as not being suitable for business (Charles & Ferguson, 2005). One of the prominent leaders of the OSS movement, Richard Stallman, left MIT in protest over increasingly restricted access to source code on government-funded computers in the MIT Artificial Intelligence Lab. He later began the free software movement by establishing the Free Software Foundation (FSF). FSF and similar non-profit organizations promoted OSS by generally adhering to their own manifestos. They criticized the traditional method of software development (e.g., formal, closed source, precise, planned) and touted the open source method of software development (e.g., informal, communal), which offered more opportunities for collaboration and was more effective by having arguably higher quality and usability (Isaacson, 2014; Roberts et al., 2006).

OSS as a process. Open source computing entails a process of creating and sharing software. This process was the essence of open source, not necessarily the software but how it is created and shared (Weber, 2004). This process freely distributed source code for others to fix, improve, and customize through a community of people, who are geographically dispersed, and who care about the code. The process, made possible by online coding platforms where individuals post their code base, allowed for code to be licensed and documented for use. OSS resided on web servers for running applications in real-time. Software development applications supported the software through continuous integration and deployment. Additionally, there were code testing tools operating simultaneously that produced multiple software updates. Building software utilizing the OSS process involved fundamentally new ways to develop software that posed a serious challenge to commercial software business dominating the software development market (Mockus, Fielding, & Herbsleb, 2002).

Research on the Federal Government's Intersection with OSS

When the commercial industry moved to CSS in the post-mainframe-to-PC era, the federal government followed suit. Exceptions occurred in computing offices and laboratories where expertise and program requirements necessitated the ability to create and customize software (e.g., space centers, energy labs). Centralized IT offices became responsible for developing, modifying, managing, and providing packaged productivity software for workers utilizing desktop computers. Beyond the skilled few in labs, centralized IT offices, and software companies, those building and sharing OSS were viewed as amateurs making a product not ready for general consumption. Moreover, software companies promoted CSS applications as being more secure, offering more features, providing better support through licensing, and easing management. Along with desktop publishing applications to support specific productivity needs,

organizations began purchasing and using more "enterprise-grade" proprietary platforms, promising a one-package software solution for all organizational functions (Balter, 2018). This resulted in government organizations focusing primarily on procuring and implementing standardized solutions or consumer-off-the-shelf (COTS) software.

In the 1990s, the federal government's reliance on CSS began to change for many reasons. The Internet arrived to server rooms and front-office desktops allowing for real-time communication and file sharing. Server and web technologies became popular; two of the fastest growing Internet products, Linux and Apache, enabled server and application customization that accommodated myriad business needs. By the 2010s, workforce skills evolved to include more advanced computing ability that permitted others outside of IT to provide computing services (i.e., digital services) and products (e.g., websites, applications, data science). This shift aligned with Holden's (2003) PAIT research: management control of IT moved to individuals outside of the CIO office.

With the maturation and use of OSS as a product for consumption in government organizations, agency instructions and executive-level policy mandated that agencies consider OSS as a viable purchasing and implementation option. With software purchasing, agencies had to consider the total cost of software ownership in meeting agency technology architecture requirements (Barrett, 2007; Burton, 2004). Because of this, the perception for OSS evolved, from skepticism about its security, reliability, and general use across the organization to belief in its efficacy as agencies used it to support critical operations (Johnson & Sears, 2017).

By the mid-2010s, the landscape of government software consisted of a mixed environment of CSS and OSS. CSS was still largely used for desktop applications and OSS for back-office technology. As the government brought in additional developers to deliver digital

services and products, these developers moved to using OSS to avoid vendor lock-in and high licensing fees, improve the quality of software being built, and support the rush to newer technologies (e.g. mobile, Linux in the data center, cloud management with open tools) (Passingham, 2016). This trend occurred outside the U.S. as well with efforts by Canadian, British, Danish, Italian, and South Korean governments created digital service and technology transformation teams ("All About Open Source", 2012; Kim & Thompson, 2013; "Open Source Software and the Public Sector", 2009).

In 2016, the U.S. government, through OMB, released the Federal Source Code Policy: Achieving Efficiency, Transparency, and Innovation through Reusable and Open Source Software, mandating executive agencies to create their own source code policy, update acquisition language to capture new custom code, and create an inventory of new custom code to post online while publishing at least 20% of the source code as OSS (Scott & Rung, 2016). The Federal Source Code Policy (FSCP) also established the Code.gov program office and corresponding technology platform. According to Jones (2018, p. 1), "the Code.gov team believe[s] in innovation and providing everyone the opportunity to perform a civic duty on a digital platform…they're passionate about making open source government projects available for all." Meanwhile, the National Defense Authorization Act for Fiscal Year 2018 required unclassified, custom developed non-defense software developed for the Department of Defense to be open source (H.R.2810, 2018).

By the late 2010s, mandates encouraged some federal government software development units to operate in the open. Units in the U.S. General Services Administration's (GSA) 18F office and the U.S. Department of Energy Lawrence Livermore National Laboratory (LLNL) were just two of many that "default to open." This meant that all source code was published

publicly in an online code sharing platform with any code not available for public release requiring appropriate justification. Those in 18F and LLNL believed this new operating posture with OSS offered many benefits. According to 18F's open source policy, "creating in-house digital services and solutions requires flexibility in how we code, with a focus on lowering costs for the American people, while improving their interactions with the U.S. government" ("18F: Open Source Policy", 2018, p. 1). Furthermore, Lee of LLNL (2018, p. 2) noted, "with open source software, agencies and contractors [could] align their goals and make the most of external resources. We [could] share projects and avoid redundancies."

Conclusion

The evolution of computing included federal government use and publication of OSS. Because of improved technology and skills along with the need to be more efficient and save money by reducing duplicative software purchasing, OSS became more common in government agencies. Executive mandates pushed for consistency among agencies to publish more OSS, although each agency had its own organizational dynamics.

The next chapter elaborates on the theoretical framework this research used to explore the organizational factors that hindered or aided agencies in publishing OSS as an indicator of their compliance with the FSCP.

Chapter 3 Conceptual Grounding

Scholarly literature that probes organizational variables that may affect OSS publishing is varied. The variables, examined in this study, (i.e., organizational culture, public engagement, structure, and organizational location) may hinder or aid federal agencies' implementation of new technology policy, as indicated by agencies' OSS publication frequency. This chapter develops the conceptual grounding for the study and describes the theoretical framework for the analysis (see Figure 3.0).

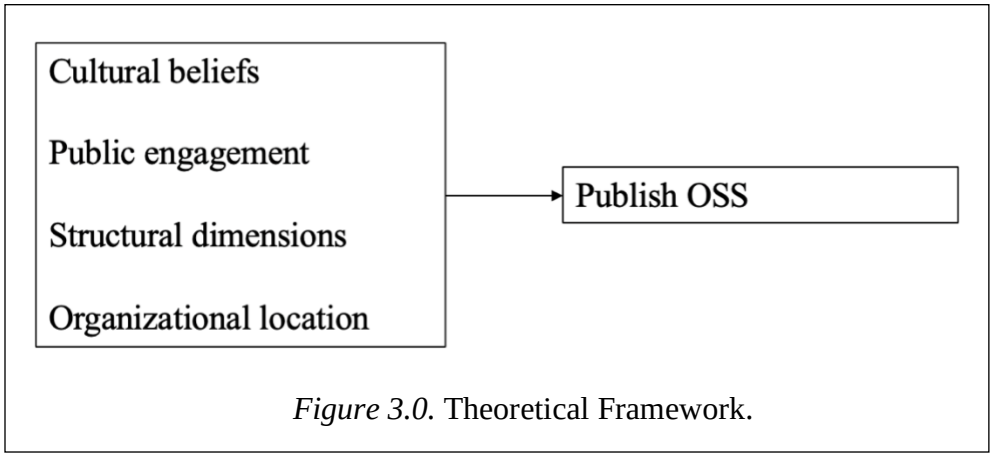


Figure 3.0. Theoretical Framework.

Research on Organizational Variables Affecting OSS Publication

This study examined four organizational variables that I hypothesized might influence the frequency of federal agencies' OSS publication. Cultural beliefs pertained to the unit's[8] actions and norms with its interaction with software development. This interaction was similar to elements of the hacker's ethos. Public engagement was the unit's ability to engage the public – meaning engagement outside its own unit; i.e., the parent organization, with another government agency, or with the general public – for communal collaboration and information sharing (e.g., policy making, code contributions). In the technology-structure perspective, varying structural

[8] See definition in Chapter 4.

dimensions (degrees of, e.g., formalization or centralization) affected how organizations responded to environmental disturbances. Because units at the boundaries of the parent organization historically have published shared OSS code, examining organizational location, where the unit is located in the broader agency hierarchy, provided insights into which units publish code publicly.

Cultural beliefs. The notion of culture, as defined by Hall (1997), was rooted in anthropological and sociological studies with a focus on relationships among small groups and tribes through the study of behaviors and norms. Culture was considered to be relatively stable and homogeneous. Most approaches to study culture began with sociologists conducting ethnographies for several decades giving way to later studies of organizational culture (Cameron & Ettington, 1988). A variety of organization theorists began to explore organizational culture as an avenue to understand social arrangements and outcomes in workplaces, which enabled managers to exercise control and ensure success (Hall, 1997; Khademian, 2002; Pettigrew, 1979). Organizational managers and scholars studied organizational culture in efforts to determine how internal and external organizational disturbances affected outcome (Hatch, 2018; Martin, 1992).

In examining MIT sociologist Edgar Schein's work, Khademian (2002) described his "theoretical framework for understanding organizational culture as three layers of organizational interaction" (p. 18). The deepest layer included the "basic assumptions [that] are taken for granted, and [are] below the level of consciousness," and that supported how members relate to the environment and each other (Khademian, 2002, p. 18). In order for organizational members to arrive at basic assumptions, they had to move through two additional layers of visible cultural manifestations. Values and beliefs consisted of what members believed "ought to be" and

included thoughts, attitudes, and philosophies. The most visible included artifacts such as language, ceremonies, rewards, experiences, and history. To understand, manage, and change culture, deeply held basic assumptions first must be recognized and changed, and these changes should reverberate through the remaining two levels.

Definition of culture. There have been numerous definitions of culture (Hall, 1997). One common definition pertaining to organizational effectiveness was a "pattern of shared basic assumptions that the group learned as it solved its problems of external adaptation and internal integration, that has worked well enough to be considered valid and, therefore, to be taught to new members as the correct way to perceive, think, and feel in relation to those problems" (Schein, 2010, p. 18). Culture was expressed through social interactions in community, facilitating sharing, and providing common understandings often emphasizing similarities for the group regarding work being done and rationales for doing it (Hatch, 2018; Khademian, 2002; Martin, 1992). In a relatively stable environment a well-established culture supported mission and success and should be nurtured and encouraged. In an environment with significant change, this same culture challenged the organization's ability to adapt to fit new organizational realities (Tunstall, 1986).

Organizations created and fostered a common culture with shared experiences through time (Schein, 2010). These shared experiences were supported by norms of behaviors based on rewards or punishment producing outcomes favorable to the organization. "Norms specify ways in which all members of the organization–or at least those in similar positions or organizational locations–[were] expected to approach their work and interact with others…and [were] generally viewed as an important component of group or organizational culture" (Cooke & Szumel, 1993,

p. 1299). Interactions or shared experiences supported ceremonies and rituals further reinforcing what an organization valued (Pettigrew, 1979).

Subculture. Organizations did not always share the same culture even though they often were treated as a monolithic phenomenon, with "one culture to a setting" (Martin & Siehl, 1983). Understanding culture called for assessing similarities and differences among units in an organization. Martin's (1992) approach for understanding culture was based on three perspectives: integration, differentiation, and fragmentation. She defined integration as the cultural manifestations that interpreted and reinforced similar themes. Differentiation meant culture manifested differently in various parts of the organization. Fragmentation occurred when ambiguity was the essence of the organization as there was not a clear culture in the entire organization or among the groups. "It [was] useful to understand the differences among perspectives and to use a multi-perspective approach or, at least, acknowledge what [was] excluded when only one perspective is used" (Martin, 1992, p. 4). However, when researchers chose to analyze an organization with all three perspectives, a greater understanding emerged.

Boisnier and Chatman (2002) found that having a differentiated subculture was beneficial to the organization. Organization subculture was a result of adaptability and agility when encountering environmental changes. This adaptability and agility potentially strengthened the organization's overall culture. In these cases, subcultures allowed for varied responses to external unit disruptions while still supporting the overall organization structure. Subcultures here were not seen as distracting from organizational culture as sometimes thought (Martin, 1992); rather, they co-existed with the overall culture (Boisnier & Chatman, 2002; Martin & Siehl, 1983).

Beliefs. In his research on beliefs, Nilsson (2014) suggested that "our beliefs constitute[d] a large part of our knowledge of the world" (p. 1). Beliefs could be used for making predictions and selecting actions and they could play an important role in understanding what is observed and enacted when confronted a particular situation. Beliefs were acquired through experience by explanations and consequences supporting or negating a belief. Beliefs supported theory-making by bolstering an epistemological position that allowed for declarative propositional statements of what ought to be. Beliefs often were tentative and changed over time. They were multifaceted: they were stated or unstated, allowing for sharing in a community (Sathe, 1983); beliefs influenced decision-making (Sapienza, 1985); they appeared in perceptual frameworks, goals, and plans (Hall, 1997); and, they were positively and negatively associated with outcomes (Nystrom & Starbuck, 1984).

Beliefs were a key component of culture, and they were embedded in the three levels of enactment that supported organizational performance. According to Schein (2010), the two lower levels of culture provided embedded organizational beliefs: those taken-for-granted supporting assumptions and those supporting rationalization, ideals, and goals. At the highest cultural level, beliefs were visibly seen through artifacts. Beliefs supported norms: "such expectations, standards, or 'norms' specif[ied] ways in which all members of the organization–or at least those in similar positions or organizational locations–[were] expected to approach their work and interact with others…and [were] generally viewed as an important component of group or organizational culture" (Cooke & Szumel, 1993, p. 1299). Beliefs were continually and collectively validated as groups tested and solved problems.

Public engagement. Public engagement was defined as an action-oriented approach to participating with individuals outside the organization. Participation included a range of

approaches to engage the public.[9] Offenbacker (2010) overviewed participatory practices and discussed why public agencies conduct participation. The six types of participatory practices organizations may employ included: 1) being part of a decision-making process; 2) representing a community-building orientation; 3) providing a form of education; 4) resolving conflicts; 5) serving as a means of collaboration; and, 6) allowing for discovery. These practices often overlapped as organizations engaged in multiple practices. For example, an organization may be involved with community building for the purpose of collective action by educating itself and others and by deliberating on varying policy options. Moreover, public agencies conducted participation for three reasons: 1) to establish and revise public policy and programs; 2) to manage public services and programs; and, 3) to enable agency-level collaboration.

Kadir (2015) suggested that the federal government has a mandate to allow for public participation. This mandate can be traced back to the Administrative Procedures Act of 1946 that required federal agencies to inform the public of their procedures and rules. The mandate included policy areas such as transportation, housing, environment, and education. Public participation traditionally involved "members of the public in the agenda-setting, decision-making, and policy-forming activities of organizations/institutions for policy development" (Rowe & Frewer, 2005, p. 253). However, the research found no general consensus on participation effectiveness, including which participation mechanisms yielded the best outcomes in specific contexts. Rowe and Frewer (2004) discussed mechanisms, but the effectiveness of mechanism varied with context, and little theory has emerged. Key works generally characterized participation or how an activity took place: participation consisted of timed-discrete events, participation often included an agenda for action, and participation created

[9] I use a broader definition of public in this study, focusing on individuals outside a unit in the same agency, individuals outside the agency but within another government agency, and the general public.

meaningful ways for individuals to participate by providing information to do so (Offenbacker, 2010).

Action-Oriented Approach to Participation. Public engagement was the act of participation. Public engagement manifested through elements of public communication, consultation, and participation (Rowe & Frewer, 2005; Rowe & Gammack, 2004). Each of these elements consisted of flows of information between the sponsor or federal program manager and public representatives or the public more generally. There were three ways for communication to occur: (1) one directional communication from sponsor to public representatives, (2) public consultation with flow of information from public representatives to the sponsor, and (3) bidirectional public participation with information going both ways from sponsor to public representative or public representative to sponsor (Rowe & Frewer, 2005).

Public engagement was often used to create effective programs. Feldman and Khademian (2007) discussed public engagement within a broader focus of inclusive management. They found public engagement often occurred in three domains: political, scientific or technical, and the local. Public engagement around these domains allowed for community building and communication where subject matter experts could use knowledge to address public problems.

Electronic public engagement. Numerous public engagement mechanisms included workshops, task forces, action planning, and citizen juries. As Internet technologies were developed and were offered commercially, new approaches to public engagement ensued as "computer-based platforms provided new opportunities for involving the public through mechanisms such as electronic consultation and virtual focus groups. Hopes for these modern solutions in the public engagement community are clearly high…[with] policy towards use of

this medium in public dealings (referred to as e-government)" (Rowe and Gammack, 2004, p. 39). E-government was a broader topic regarding how the government worked with new electronic tools, while OSS public engagement created a process for information to flow between an agency and the public that drove involvement and established community around a public program and software.

Structural dimensions. According to Scott and Davis (2007),

> Most analysts have conceived of organizations as social structures created by individuals to support the collaborative pursuit of specified goals. All organizations confront a number of common problems: all must define (and redefine) their objectives; all must induce participants to contribute services; all must control and coordinate these contributions; resources must be garnered from the environment and products and services dispensed; participants must be selected, trained, and replaced; and some sort of working accommodation with the neighbors must be achieved. (p. 11)

Early organizational research began in industrial society with organizations as collective entities pursuing specific goals through formal social structures (Hatch, 2018; Scott & Davis, 2007). Rational systems theorists noted that rules were of the utmost import and that individuals should be bounded. An early and prominent example of rational systems theory was Max Weber's (2017) ideal type bureaucracy with its fixed division of labor, clearly defined hierarchical offices, appointment of technical experts, fixed salaries, establishment of careers, promotion based on seniority, official work separated from ownership, and the understanding that rules govern the office. This view of organization has been scrutinized broadly but many still see it as effective for many large organizations in 2019 (e.g., government, corporations, churches, academic institutions). However, Weber (2017) also outlined where bureaucracies would be

inappropriate: 1) when organizations were small and direct supervision was not needed; 2) when the environment was unstable due to constant changes; and, 3) when the organization employed a large number of professionals who were highly trained and expected a high standard of performance.

Open systems theorists were at the other end of the organizational analysis spectrum in their consideration of environmental influences affecting the organization. Structural contingency theory began with the notion that organizations structure according to environmental demands (Lawrence & Lorsch, 1967). In stable environments, "the mechanistic form of organization work[ed] best because of the efficiencies it [could] generate using standard procedures to perform routine activities" (Hatch, 2018, p. 82). Further, the organization optimized routine activities, which allowed for gains in performance and reduced transaction costs. "Organic organization [was] more effective under conditions of rapid environmental change, because this organizational form support[ed] the flexibility needed for…adaptation" (Hatch, 2018, p. 82). Lawrence and Lorsch (1967) conducted a study of the U.S. chemical industry and found that groups performing more routine tasks tended to be more formally structured than those conducting more open-ended and complex tasks.

Organizations are a product of the time and place in which they were established. Rational systems generally aligned with the structures and functions of large bureaucratic organizations in industrial society. However, this organizational form may not be practical in 2019 when more open system structures were needed. These forms were contingent upon the environment and called for rapid adaptability particularly with technological change. Open-ended and complex tasks required newer forms of organizing with less hierarchy, formalization, and centralization where units on the "front-lines" could make decisions that best supported their

work, including operating in a contributory and collaborative manner. As Germain (2014, p. 6) noted, "the government has a strong, top-down, command-and-control structure that is the exact opposite of how open source works."

Technology-structure literature. One way to examine organizational structure was through an open system technology-structure approach. Technology,[10] or the core technology of an organization, was the process of taking inputs and creating outputs in the form of a product or service. Technology encompassed the "activities or processes that comprised the methods of production" (Hatch, 1998, p.128). Technology was the functional apparatus of the organization (e.g., marketing, HR, manufacturing), and it could be examined at several levels of analysis (e.g., environment, organization, department, sub-organizations, task), thus providing a way to compare functions of organizations (e.g., HR and marketing) and across organizations (e.g., HR in two government agencies).

Technology required a complementary organization. Woodward (1981) conducted a systematic study of organizational structure and technology by empirically examining the relationships between technology and social structure in the 1950s in English factories. She found organizational characteristics, technology, and success were linked together, facilitating the formation of a particular organizational unit most appropriate given the system of production. Further, when considering individuals of the unit, she found when the work routine was high and technical complexity was moderate, assembly workers performed optimally for mass production (Woodward, 1981). Conversely, engineers operated best when routineness of work with continuous processing of products was low and the complexity of technology was high.

[10] In this study, technology refers to tasks associated with implementation of policy such as agency publication of OSS.

Based on a meta-analysis of the technology-structure literature, researchers have concluded that organizations often respond to environmental disturbances and environmental influences in predictable ways (Fry, 1982; Scott & Davis, 2007) (See Figure 3.1).

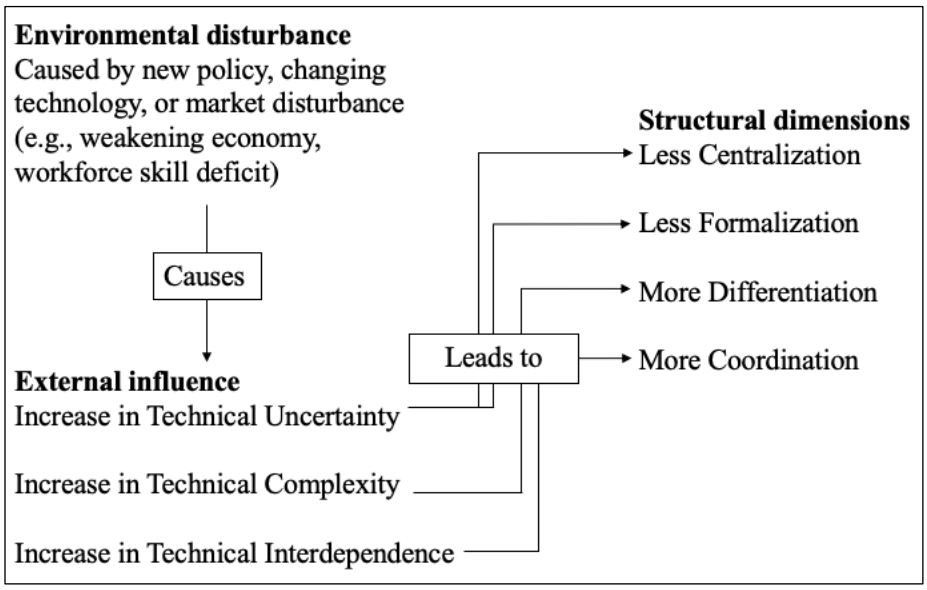


Figure 3.1. Technology-Structure Effects on Organizations

Environmental disturbances produced external influences including technical uncertainty, technical complexity, and technical interdependence, leading to various effects on four structural dimensions – centralization, formalization, differentiation, and coordination. Each of these dimensions appears in the theoretical framework for this study.

Scott and Davis (2007) discussed research on each influence and dimension. Their findings are summarized below with additional supporting research.

- Technical uncertainty emerged under conditions of technical complexity and high rates of change. Environmental changes included the organization's competitive situation, the impact of new technology, and volatility in the market. Technical

uncertainty increased as organizations were mandated to implement new technology, especially when an organization has minimal experience with the technology.

- Technical complexity referred to an organization's ability to process information about the implementation of a new technology. Organizations encountered difficulty implementing new technology when information regarding the technology was uncertain due to a rapidly changing environment or overwhelming amounts of information. A rapidly changing environment and little information could have similar effects, if an organization did not have enough information to implement a new technology.
- Technical interdependence referred to a unit's work processes that were interrelated in the unit and with units in the broader organization. Thompson (2017) defined three types of interdependence: pooled, sequential, and reciprocal. These affected the organization's ability to implement a technology based on the levels of coordination needed from other technical and functional units.
- Formalization included the organization's use of written rules, job descriptions, procedures, and communications. Units within organizations performing routine tasks (e.g., production, administration) tended to be more formalized than units performing more complex and open-ended tasks (e.g., research and development). Formalization was an attempt to make behavior predictable through explicit procedures (e.g., rules, standard operating procedures). In contrast, the lack of formalization allowed for unit flexibility and spontaneity (Hatch, 2018; Scott & Davis, 2007).

- Centralization described where decision-making took place in an organization. In more centralized organizations, decisions were made by those at the top of the overall hierarchy; while, in more decentralized settings, many organizational participants from differing levels and parts of the organization were involved in decision-making. "The choice of which functions or activities [were] to be performed and controlled by a 'headquarters' unit [was] the essence of the decentralization-centralization problem. The more these activities [were] allocated to each subunit, the greater its autonomy, and the more decentralized the organization" (Mackenzie, 1978, p. 201).
- Differentiation entailed the separation of work by expertise or the diversity of skills within and across units. Skills may be dispersed across units by "products produced, services provided, clients served, processes used, or location as seen in product teams, cross-functional workgroups, and task forces" (Hatch, 2018, p. 112).
- Coordination was the information exchange accomplished among units through communication. Communication was greater in rapidly changing environments when the unit was less complex, less formal, and more decentralized (Bacharach & Aiken, 1977; Hage, Aiken, & Marrett, 1971).

Organizational location. The organizational design literature provided a foundation for understanding a unit's ability to respond to environmental disturbances based on where the unit was located in the overall organization.[11] According to Scott and Davis (2007), "in response to greater amounts of task complexity, uncertainty, and interdependence, organizational forms

[11] Organizational location (or position in an overall organizational hierarchy) is a structural dimension frequently associated with centralization. The technology-structure literature focuses on centralization of decision-making with less attention to hierarchy. I looked at this dimension apart from decision-making in order to focus on units inside and outside the agency CIO office responsible for FSCP implementation, because OSS historically has been developed and published in units at the boundary of the organization (e.g., labs, space centers).

[were] likely to exhibit increasing differentiation and structural flexibility" (p. 124). They provided multiple forms of organization. The "first three forms – simple, bureaucracy, and functional – presume[d] singularity of purpose and unity of command" with rational system characteristics (Scott & Davis, 2007, p. 124). Newer forms of organizing with open systems included divisional, matrix, adhocracy, and networks, accommodating multiple objectives and divided authority.

The organization design literature focused on structuring for optimal coordination and interactions among activities and units. The structure of an organization was defined "simply as the sum total of the ways in which its labor [was] divided into distinct tasks and then its coordination [was] achieved among these tasks. The elements of structure should be selected to achieve an internal consistency or harmony, as well as a basic consistency with the organization's situation" (Mintzberg, 1983, p. 20). A number of structural configurations with various parameters of structural design emerged, predicated upon labor (Heydebrand, 1989), job specialization (Mintzberg, 1983), coordination (Harris and Raviv, 2014), and tasks (Dixit, 2012).

New technology arguably required new forms of organizing. The U.S. federal government frequently relied on bureaucratic organizational forms. Divisions were organized in hierarchical fashion with a command-and-control structure for goal and rule management. Specifically, units exhibited several rational organizational form characteristics: routine tasks, high levels of formalization and centralization, and line departments directly supporting product development. Government units producing code in 2019, exhibited open system organization forms similar to the collegial structuring of units in the early days of government computing. Divisions operated relatively autonomously, organized work by function or project criteria,

organized teams to produce a particular product, had low formalization and centralization, and conducted activities across boundaries (Beyer, 2009; Feld, 2016; Levy, 2010).

Conclusion

This chapter identified organizational factors that may either aid or hinder agencies' ability to implement new technology. Their analysis provided insight into whether and how agencies publish OSS. Organizational factors including cultural beliefs, public engagement, structural dimensions, and organizational location are integral to agencies' ability to be responsive to environmental disturbances. Chapter Four describes the research design for examining these factors and their associations with publishing OSS.

Chapter 4 Research Design and Methodology

This study examined whether and how the organizational features of federal software development units is related to the units' frequency of publishing OSS, a key indicator of compliance with the FSCP policy compliance. The chapter begins by describing the approach to the research, including its methodological orientation and my epistemological grounding. Next it explores processes of data collection and analysis. The chapter concludes with a discussion of the difficulties encountered during the research.

Approach to the Research

I used qualitative methods to determine whether and how organizational factors affected units' OSS publication frequency. Specifically, I followed Creswell's (2009) common aspects of qualitative research:

- *Collected data in the natural setting* by focusing on software development activities occurring in federal agencies;
- Collected and examined primary data as the *key instrument of the study*;
- *Collected data from multiple sources* including metadata, interviews, and artifacts;
- Used *inductive data analysis* following grounded theory for coding, categorizing, and memoing for theory building;
- *Focused on participants' meanings* with effort to reduce bias and maximize search for rich meaning in the data;
- *Used an emergent design* with data collection and analysis; and,
- Created a *holistic account* of the phenomenon.

Orientation

I employed a grounded theory (GT) approach. Corbin and Strauss (1990, p. 5) described this as an approach "to develop a well-integrated set of concepts that provide a thorough theoretical explanation of social phenomena under study." In such an approach to theoretical explanation of a social phenomenon, "data [were] systematically gathered and analyzed [and] theory evolve[ed] during research…through continuous interplay between analysis and data collection" (Strauss & Corbin, 1994, p. 273). I utilized GT for this qualitative inquiry as no theory was prevalent, and I sought to understand an emerging phenomenon; it provided a process for allowing the data to tell the story as I systematically gathered and organized data and interpreted findings. A major strategy for furthering the discovery of grounded theory in research was the technique of comparative analysis (Glaser & Strauss, 1967). Comparative analysis encompassed gathering data, analyzing data with coding (e.g., open, selective) to generate conceptual categories, developing analytic memos allowing for theory building and evaluation, and generating new questions for in-depth theoretical sampling, and ultimately, theory generation.

Glaser and Strauss developed GT in 1967 as a general approach to comparative analysis to generate theory from empirical data as opposed to other approaches where data was collected and analyzed to validate or test an existing theory. The central role of core categories was fundamental to the emergence of GT. Core categories are developed by open and axial coding leading to selective coding and possible theory development. GT has become controversial among qualitative researchers because of the two diverging approaches offered by its founders.

The Glaser approach consisted of elements from the initial work of Glaser and Strauss (1967), and later, Glaser alone (1978). It featured a systematic approach to analysis with various

techniques, including sampling, coding, memoing, integrating, and writing. Glaser called for creativity and flexibility without describing precisely how the practices need to be employed. He also believed the researcher should set aside preconceived ideas about the topic and learn from the data while not relying on standard precepts of established research procedures (i.e., literature review, theoretical framework).

The Straussian approach followed the initial GT techniques of Glaser and Strauss (1967) but offered a different approach with data collection and theory building as noted in later works by Strauss (1987) and by Strauss and Corbin (1990). The Straussian approach emphasized the need for structure to aid with analysis, particularly with the coding schema during axial coding by focusing on conditions, interactions, strategies, tactics, and consequences. His approach was much more prescriptive, limiting the researcher's creativity and flexibility. Moreover, Strauss insisted on structure in advance of data collection (i.e., literature reviews, theoretical frameworks), and this approach suggested the researcher's experience, knowledge, and interest were an important part of the initial focus of the study.

Although both GT founders used the same tools for data collection and analysis – coding, memoing, integrating, and writing, the structure of the approach differed. The main differences were using precepts to research (i.e., literature reviews, theoretical frameworks) while incorporating research experience, knowledge, and interest. Furthermore, the more prescriptive Straussian approach called for a specific way to conduct axial coding for comparative analysis. Some scholars argued that because the Straussian approach diverged from the Glaser approach, mostly because it relied on precepts, it led to a fundamental change in the meaning of GT. Regardless of these differences, these approaches are useful and valuable qualitative research methods (Gibson & Hartman, 2014; Holton & Walsh, 2016).

This study used the Straussian approach. I came into the study with my own experience, knowledge, and interest in technology policy implementation and OSS publishing. For proposal development, I initially researched the topic to determine an approach to study the phenomenon in question by conducting a literature review, developing a theoretical framework, and generating research questions based on organizational factors hypothesized to affect OSS publication. I also collected metadata from the GitHub API to determine which federal agencies were publishing OSS. Finally, I used emergent data analysis techniques provided by Glaser and Strauss (1967), including coding, memoing, and writing, and ultimately generating a theory of implementing technology policy, focusing on OSS publishing as viewed by participants in the sampled units.

Researcher's Epistemological Grounding

My professional background in software development and my role in the federal government influenced this research and its outcomes. I am a federal employee with 15 years of government IT experience, and I am employed at one of the 24 CFO Act agencies included in this study. I have extensive experience and education in software development and government. I worked in the USCIO unit in OMB to develop supplemental guidance to the Open Data Policy (M-13-13) and to agency IT policy and program audits. I later served as the GSA's open source liaison responsible for agency compliance with the FSCP. In this position, the USCIO asked me to become the Director of Code.gov. As Director of Code.gov, I am responsible for leading a team to build the Code.gov technical platform, ensuring CFO Act agency compliance with the FSCP (M-16-21), and open sourcing government software.

I understand participants may have provided responses to the interview questions based on their views of my government and academic roles. I hoped to minimize biased participant

responses, calm hesitation, and build trust by disclosing the purpose of study and by being transparent with what I would do with the data collected, including ensuring participant and agency confidentiality.

My bias is towards using and publishing OSS, because I find it beneficial and effective for reducing duplicative purchasing of software, promoting open collaboration and sharing for creating software that ultimately allows for creating better code.[12] The rationale for this research originated from my desire to understand why some agencies implement technology policy and the FCSP, while others struggle to publish OSS as mandated in the FCSP. I sought to minimize bias by utilizing qualitative methods with multiple data sources to create a more holistic and robust picture of government technology and policy implementation.

Data Collection

Data collection began immediately following study approval from my university's Institutional Review Board (IRB).[13] I began with a pilot to test the data collection and analysis processes with individuals from the Code.gov community. At the same time, I conducted a metadata analysis of CFO Act agencies listed on GitHub.com. Then, I recruited participants from CFO Act agencies using the Code.gov listserv who met specific criteria relating to software development activities. Finally, I collected data through elite semi-structured interviews and from artifacts.

Pilot. Before beginning the full qualitative data collection and analysis effort for the study, I conducted a pilot by enlisting five Code.gov community members. The steps of the pilot mirrored those of the study – coordinating interviews, sending formal invitations and consent

[12] Software consists of source code or simply "code" as a collection of files with text commands to be executed as an application on a server, computer, or technology device.
[13] Appendix removed for privacy. Available upon request.

forms, and audio recording and transcribing. This allowed me to solidify the processes of the study and obtain feedback from the pilot participants. Research has shown pilots allow the researcher to try out strategies, foreshadow research problems and questions, highlight gaps in data collection, and help the researcher to consider broader and significant issues including research validity, ethics, and representation (Marshall & Rossman, 2011). Additionally, the pilot assisted with refinement of the interview guide and transcription process. During the pilot, I used open source Python scripts and the Google Speech-to-Text API to automate a portion of the transcription process.

Metadata collection. To complete the metadata collection, I used the GitHub, Inc. (GH) GraphQL API ("GraphQL API v4: Overview," 2018). GH collects metadata of coding activities occurring through its website with code repository activities consisting of forks, stars, commits, pull requests[14] (PR), and issues, and provides these metadata free to registered GH users ("GitHub is How People Build Software," 2018). Moreover, GH associates coding activity with user and organization profiles. GraphQL is a program developed by Facebook, Inc. and adopted by technology companies for querying, adding, modifying, and deleting data in a database. An API is a program allowing for the transfer of data over Hypertext Transfer Protocol (HTTP).

GH maintains a government site ("How Agencies Build Software," 2018) and corresponding code repository ("github/government.github.com," 2018) with a publicly curated and maintained list of government organizations worldwide. I selected the sub-list of federal organizations and manually verified each organization belonging to a CFO Act agency.[15] With this updated organizational list, I used the API to find the organization code repositories of all

[14] A pull request allows users to submit modified code to the original owner's source code for review and inclusion in the original code base.
[15] In November 2018, there were 136 federal government organizations belonging to 23 of 24 CFO Act agencies on GitHub.com.

CFO Act agencies. I enhanced the open source Python scripts developed by Irizarry (2018) to generate the code repository list for all CFO Act agencies. These scripts used the GH API with GraphQL commands to develop the sampling frame of federal CFO Act agencies on GH with their associated code publishing frequency.[16]

The metadata sampling frame included 23 of 24 CFO Act agencies using GitHub.com.[17] Using the metadata, I divided the agencies into thirds based on their OSS publishing frequency (i.e., publishing minimally, intermediately, and frequently) from the total number of code repositories created before and after the release of the FSCP in August 2016.[18] This sampling frame showed federal government code repository activity by agency and by code publishing frequency both before and after the FSCP mandate. This sampling frame indicated general agency progress implementing the FSCP.

Participants. To gather a representative sample by agency OSS publishing frequency, I emailed the Code.gov's community-of-practice listserv consisting of approximately 670 federal, military, and government-contracted employees interested in software development or working in a federal software development organization. The email included the following criteria for voluntary participants: they needed to 1) be a federal employee in one of the 24 CFO Act agencies; 2) develop code or manage federal and contract employees who develop code; and 3) manage or work in a software development unit, not necessarily in an IT office, but within a unit responsible for developing software.

[16] This step was completed in November 2018 with results located in Appendix B: GitHub.com Metadata of Open Source Software (OSS) Published by CFO Act Agency.
[17] The Nuclear Regulatory Commission (NRC) does not have any organizations or code repositories on GitHub.com.
[18] Located in Appendix B: GitHub.com Metadata of Open Source Software (OSS) Published by CFO Act Agency.

Many units conducted software development activities across all CFO Act agencies. A "unit" for this study was defined by formal organizational structure (i.e., organizational chart) or by a project. Participants included executives who managed the activities of multiple divisions; division directors at the GS-15-level, who focused on software development activities in their division; branch chiefs at the GS-14-level, who lead the software development activities in their branch; and project team members at the GS-13 and the GS-14-level, who focused on activities associated with a particular project. For example, one participant was a CIO of a research lab, another was a division director, and a third was a data scientist on a specific project with other data scientists. In this study, all participants were responsible for units developing software.

I initially received a low response rate from agencies with lower publishing frequency, which prompted me to send direct emails to underrepresented agency contacts asking for volunteer participants or to forward invitations to others who might meet the participant criteria and be willing to participate. I sent direct emails to agency FSCP policy liaisons (those responsible for implementing the policy at their agency), personal network contacts within the Code.gov community-of-practice, and individuals in my professional network including agency CIOs, directors of software development, and developers. In all, participants from 20 of the 23 agencies listed on GitHub.com agreed to participate, resulting in 25 individual participants. There were four agencies with multiple participants, but the participants worked in separate software development units (n = 25, see Table 4.0).[19] I chose volunteers to participate based on uniqueness of their agency and timing of receipt of their willingness to participate, which was

[19] Although the sample includes representatives of 25 units, the data are from individual participants from those units. Throughout the document, when reporting findings, the participants are speaking for their units.

done on a "first-come, first-serve" basis. I then formally invited participants to the study with a Recruitment Email[20] and Consent Form. [21]

Table 4.0

Participants in Study

Agency Publishing Frequency	Agencies	Participants
Minimally	6	7
Intermediately	6	7
Frequently	8	11
Total	20	25

The initial publishing frequency categories were derived with public data and divided evenly between categories by thirds.[22] Most participants in the sample were from units in agencies that the metadata determined were frequently publishing OSS (see Table 4.0); however, more participants in units self-reported as minimally publishing, demonstrating that participants' perceptions of publication frequency of the unit was often different than what was reported for the agency on GitHub (see Table 4.1).

Table 4.1

Publishing Frequency by Unit

Unit Publishing Frequency	Participants	% Sample
Minimally	14	56%
Intermediately	5	20%
Frequently	6	24%

The categories for unit OSS publication were subjective as there were no clear quantitative boundaries; what one participant called "frequent" OSS publication, another may categorize as "intermediate." That more participants in the sample reported their units "minimally" published OSS potentially suggested an unbalanced reflection of the phenomenon.

[20] Appendix removed for privacy. Available upon request.
[21] Appendix removed for privacy. Available upon request.
[22] Located in Appendix B: GitHub.com Metadata of Open Source Software (OSS) Published by CFO Act Agency.

The sampling included roughly equal numbers of units across the OSS publishing categories (6 minimal, 6 intermediate, 8 frequent), even if the self-reported use was skewed. The qualitative nature of the study evidently offered a holistic view of the data. Furthermore, the use of qualitative methods created in-depth, rich descriptions for better understanding and generalization of the phenomenon (Marshall & Rossman, 2011).

Based on an examination of the data, I delineated the units' functions into seven categories (see Table 4.2) with data science (28%) and software development (44%) being most common.

Table 4.2

Unit Functions by Percent

Unit Function	Participants	% Sample
Business Analysis	1	4%
Data Science	7	28%
Policy Implementation	2	8%
Research	2	8%
Software Development	11	44%
Technology Deployment	1	4%
Web Development	1	4%

More non-software-related units (i.e., Business Analysis, Data Science, Policy Implementation, Research, Technology Deployment) (46%) than software-related units (i.e., Software Development, Web Development) (42%) participated in the study and these non-software-related units published more often (see Table 4.3).

Table 4.3

Publishing Frequency by Unit Function

Unit Type	Participants	% Sample
Software	12	42%
Non-Software	13	46%

Units consumed OSS more frequently than they published: with frequently consuming units (52%) only minimally publishing OSS (24%) (see Table 4.4).

Table 4.4

Units Consuming Compared to Publishing OSS

Unit Publishing Frequency	Consuming	% Consuming	Publishing	% Publishing
Minimally	7	28%	14	56%
Intermediately	5	20%	5	20%
Frequently	13	52%	6	24%

This disparity among units consuming and publishing OSS was expected, since government agencies' consumed OSS well before the Internet became a common part of computing in most offices (Germain, 2014). The appetite for government consumption of OSS has evolved into general use across agencies as they employed it to support "mission critical" applications (Johnson & Sears, 2017).

Interviews. "Interviewing can inform us about the nature of social life" (Weiss, 1994, p. 117). Interviews are common in qualitative research when the study phenomenon is complex and not easily understood. They provide a mechanism for "discoverability" and richness of first-hand accounts of events and processes through participant understanding and interpretation. Aberbach and Rockman (2008) stated that "interviewing is often important if one needs to know what a set of people think, or they interpret an event or series of events, or what they have done or are planning to do" (p. 673).

For this study, the interview strategy consisted of elite interviewing with open-ended questions via a semi-structured interview guide.[23] Elite interviewing involved interviewing individuals based on their professional role, who are most closely associated with the phenomenon in a field setting. Participants in this study were individuals involved with software

[23] Located in Appendix C: Interview Guide.

development activities in their agencies. A semi-structured interview method with open-ended questions allowed for data collection and interpretation of the richness of the phenomenon. Each interview lasted approximately one hour via a video conferencing tool (e.g., Zoom, Google Meet). I audio recorded the interviews and later transcribed and validated the transcripts with member checking.

Artifacts. Over the course of the study, I collected a variety of documents, including agency organization charts, open source policies, internal open source and coding instructions, acquisition guidance to capture code developed from contractors, and procedures to open source. I requested documentation related to software development with coding, design, and testing as this offered insight into coding practices within the organization. I collected other forms of documentation such as, personal correspondence, memos, and notes. This multitude of documents helped to further create a holistic picture of organizational activities. Documentation was public and private, and care was taken to protect the confidentiality and security of the documents. No agency or identifiable participant information will be released without written participant consent.

Data Analysis

I conducted qualitative data analysis (QDA) using GT. Grounded theory involved an interpreted recursive process where the researcher continually conducted theoretical sampling by analyzing data and generating conceptual categories through data comparisons and development of theoretical statements. According to Locke (1996), GT featured "two key analytic operations that occur in tandem: making constant comparisons and theoretical sampling" (p. 240). Constant comparisons of data occurred as soon as they were collected. Shortly thereafter, theoretical sampling began as emergent theory drove ongoing data collection. This process continued until

theoretical saturation or analysis yielded “no new examples and properties of a conceptual category” (Locke, 1996, p. 241).

Data collection occurred in four phases from May to August 2019. The first phase was the pilot, including five participants, phase two included 11 participants, phase three nine participants, and phase four five participants. The phases were separated this way based on participant availability, with data collection taking approximately two weeks and analysis taking roughly two weeks with frequent overlap through the month. As interviews were completed and artifacts collected, I organized and prepared the data for analysis.

Organizing the data included assigning a unique identifier for each participant comprised of unit frequency of publication and participant number (i.e., F-1 means the first participant from a unit frequently publishing OSS). This unique identifier helped with confidentially and with tracking participants through the study and later in this document. Analysis included transcribing interviews, member checking for accuracy, organizing artifacts, and importing all data into NVivo,[24] a QDA software program. Once I entered data in NVivo, I read through the data to obtain a general sense of concepts prior to coding. The goal here was not to arrive at one complete set of data for the whole study before beginning analysis, but to use the recursive analytic GT approach to inform each new data collection activity throughout the study. For Strauss (1987), “data collection never entirely cease[d] because coding and memoing continue to raise fresh questions that can only be addressed by the gathering of new data or the examining of previous data” (p. 27). At this point I also created memos about the data collection and analysis processes that included initial thoughts about findings, such as data categories, overarching themes, and considering level of specificity for further questioning.

[24] https://www.qsrinternational.com/nvivo/home

"By breaking down and conceptualizing we [meant] taking apart an observation, a sentence, a paragraph, and giving each discrete incident, idea, or event, a name, something that [stood] for or represent[ed] a phenomenon" (Strauss & Corbin, 1990, p. 63). This process led to open coding or labeling the phenomenon by making comparisons of the data. Further, open coding analysis allowed me to develop categories by recognizing properties and dimensions of the main codes. I generated codes and categories through an iterative, recursive process with the raw data. This further informed memoing, writing up code notes, and creating conceptual labels.

The next step was axial coding, which included "a set of procedures whereby data [were] put back together in new ways after open coding, by making connections between categories" (Strauss and Corbin, 1990, p. 96). I conducted axial coding by examining conditions, context, action and interaction, and consequences, which ultimately resulted in further refinement of the initial categories developed with open coding. Open coding provided a method to parcel categorical concepts by properties and dimensions, where axial coding allowed me to go further with analysis by grouping the data together through making connections between categories and sub-categories. While doing analysis, I alternated between open and axial coding.

Throughout open and axial coding for category creation, I used memoing, including code notes, theoretical notes, and operational notes to assist with research organization, theoretical sampling, and generation of theoretical statements. Code notes consisted of conceptual labels, paradigm features, and indications of processes. Theoretical notes allowed for theoretically sensitizing and summarizing findings from the data through GT procedures (Strauss & Corbin, 1990). Operational notes contained directions for processes and procedures I used in data collection and analysis for reference.

Finally, in using theoretical sampling I decided "which additional data (events, activities, populations, etc.) [were] relevant to explicate and develop all properties of the evolving conceptual categories" (Locke, 1996, p. 240). This sampling was based on the concepts that offered theoretical relevance to the emerging theory. Through theoretical sampling and data analysis with coding, I generated theoretical statements to analyze the data. This continued until theoretical saturation. At this point, I utilized selective coding or "the process of selecting the core category, systematically relating it to other categories, validating those relationships, and filling in categories that needed further refinement and development" (Strauss & Corbin, 1990, p. 116). Selective coding provided the basis to create the overarching theory of the study.

Reliability and validity. While performing data analysis, I considered issues of reliability and validity. Qualitative reliability ensures that "the researcher's approach is consistent across different researchers and different projects" (Creswell, 2009, p. 190). Reliability includes following a recognized qualitative research approach, checking for obvious mistakes in transcripts, and adhering to the definition and meaning of codes as to prevent code drift.

Qualitative validity ensures "corroborating evidence [is] collected through multiple methods – interviews and documents – to locate major and minor themes." (Creswell, 2009, p. 190; Creswell & Miller, 2000, p. 127). For this study, validity included 1) triangulation of data by using elite interviews and artifacts to corroborate evidence and validate a holistic view of the phenomenon; 2) member checking by providing transcribed interviews to participants for further clarification and reduction of researcher bias; and, 3) using thick descriptions with analytic memos as data was collected to ensure understanding with thematic narratives.

Difficulties Encountered in Doing the Research

Difficulties occur when conducting primary research, and this study was no different. Beyond the normal challenges of commencing a large study requiring a significant amount of time, scheduling interviews with participants who had busy work schedules and applying the principles of how to do primary research for the first time were two main difficulties encountered in conducting this research. The first pertained to securing participants for the study. The initial emails to the Code.gov community-of-practice listserv resulted in a number of volunteers skewed toward agencies with higher OSS publishing frequencies. This could have indicated that agencies that conduct software development understand the importance of the mandate of the FSCP, and were generally more willing to work outside the boundaries of their own organization to share and learn in a community. These notions were evident in the data and are discussed in later chapters. To mitigate this difficulty, I contacted members of my professional network directly and then "snowballed" the request by asking people I knew in the Code.gov community to pass along the invitation to others, which eventually led to a representative sample. The second difficulty was likely something all researchers and writers face as they work on a study of this magnitude and that was the amount of time and effort it took to collect and analyze data. Applying the research principles for how to collect data, analyze it with GT method, record, transcribe, code, memo, etc. involved an enormous amount of analytical power over many months in examining over 1,500 data points.

Conclusion

A qualitative approach relying on grounded theory enabled this exploration of organizational factors affecting agencies' ability to implement the FSCP as demonstrated by their frequency of publishing OSS. Initial precepts of the study included a literature review for

conceptual grounding, development of a theoretical framework, and creation of general research questions. I drew a representative sample of agencies and individuals and then collected and analyzed the data with triangulation based on member checking and artifacts. The next chapter presents the findings of the study.

Chapter 5 Results

This study examined whether and how organizational factors impacted the frequency of OSS publication by federal software development units. This chapter presents findings on the relationships between a unit's publishing frequency and its cultural beliefs, public engagement, structural dimensions, and organizational location. Discussion begins by reviewing the theoretical framework.

Variables and Category Generation

As Chapter 3 elaborated, variables drawn from organizational theory including cultural beliefs, public engagement, structural dimensions, and organizational location and explored their relationships with agency units' frequency of publishing OSS. Using grounded theory in examining the data analysis, I generated categories for each of the variables (see Figure 5.0).

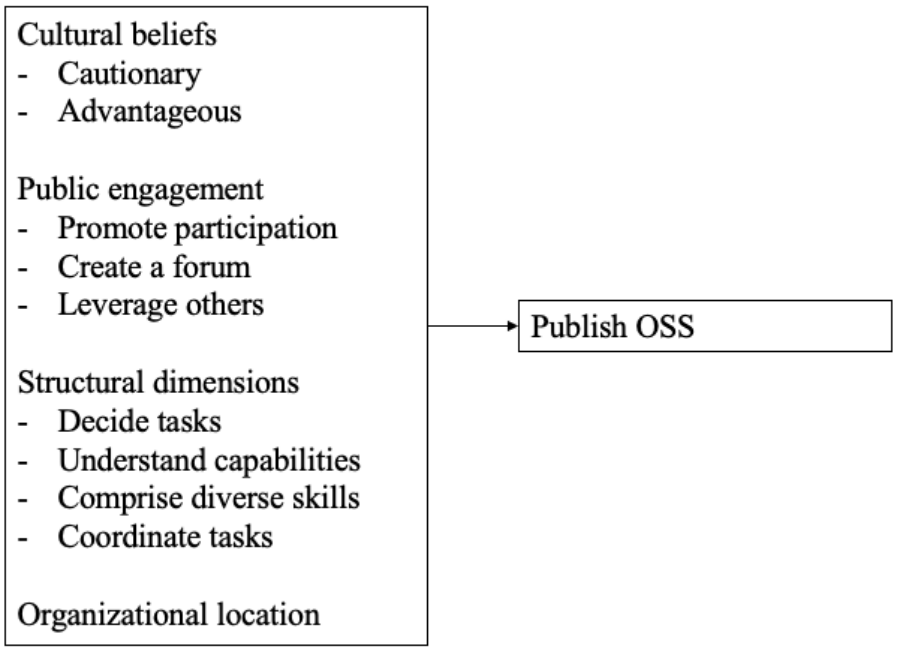


Figure 5.0. Expanded Theoretical Framework with Grounded Theory-Developed Categories.

Cultural Beliefs

The first research question asked whether and how do a unit's cultural beliefs affect its publication of OSS? Cultural beliefs consisted of multiple theoretical categories and sub-categories (see Figure 5.1).

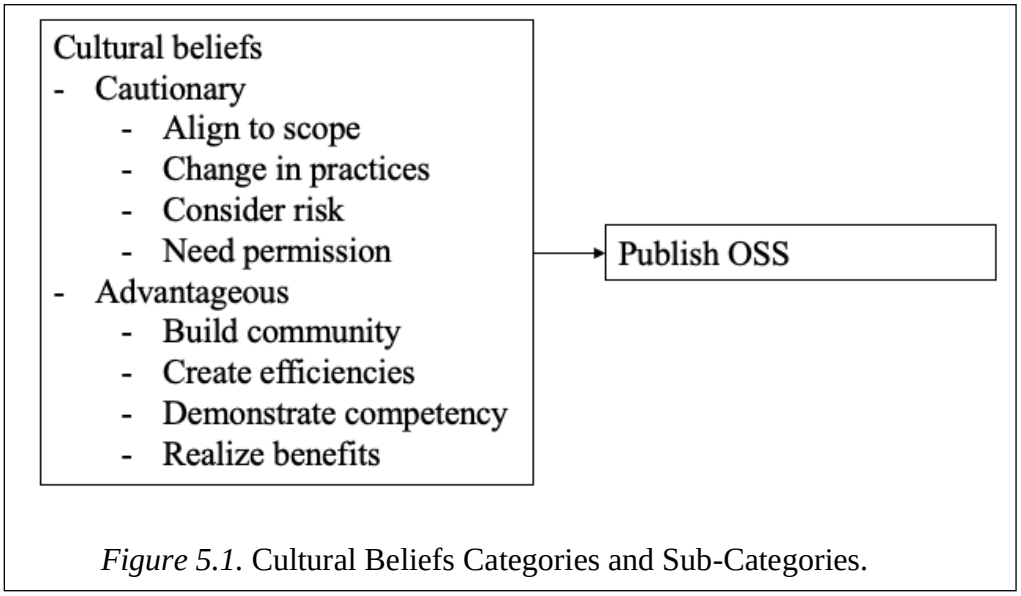


Figure 5.1. Cultural Beliefs Categories and Sub-Categories.

As a first step to understanding a unit's beliefs regarding OSS publication, I provided each participant their agency's OSS repository count and release frequency. When asked whether they believed their unit released an "appropriate amount of OSS," participants were almost evenly split, with 48% believing their unit released the appropriate amount and 44% who did not, with two participants (8%) not commenting. Of those participants in units that minimally published OSS, more believed their unit was not publishing enough OSS (M = 9) than those who thought they published an appropriate amount (M = 4) (see Table 5.0).

Table 5.0

Participant Beliefs on Amount of OSS to Release

Appropriate Amount	Count	% Sample	Frequency of OSS
Yes	12	48%	F = 5, I = 3, M = 4
No	11	44%	F = 1, I = 1, M = 9
Unknown	2	8%	F = 0, I = 1, M = 1

Note. OSS publishing frequency: F = Frequently, I = Intermediately, M = Minimally.

The variation between those who believed their units published enough and those that did not was minimal, requiring additional probing. Those participants who believed their units should publish OSS more frequently were in units that minimally published OSS, suggesting that other factors could be hindering their ability to publish OSS. The data further supported the generation of two categories pertaining to beliefs – Cautionary and Advantageous. Although the data supported the generation of these categories, some participants' beliefs fit both categories.

Cautionary. Participants whose units held cautionary beliefs were reticent in their response to new technology and technology policy. This reticence evidently stemmed from the inherent scrutiny government entities receive through annual ethics reviews and training as well as frequent public inquiry: "we don't want to be the system that the *Washington Post* reports got breached" (M-2). All units whose respondents were in this category minimally published OSS (52%) (see Table 5.1). Moreover, these participants believed OSS publication was not in scope with their daily work, was negatively associated with change, created risk, and required permission to publish.

Table 5.1

Participants in Units with Cautionary Beliefs

Unit Publishing Frequency	Participants	% Sample
Minimally	13	52%
Intermediately	0	0%
Frequently	0	0%

Align to scope. This sub-category included data from participants who believed that software development should be aligned to a specific organizational scope. Participants believed companies that set out to build technology should open source software but the government should not: "some companies [in private industry], [are] responsible for developing technology products [that] are inherently technical and invent new software. IT [in government] is identifying the right solution and weaving it into the organization" (M-9). Because participants thought their units focused on a broader scope of work (e.g., law enforcement, cybersecurity, natural resources), they believed the units in these instances should not develop software. Furthermore, these units concentrated their IT efforts on analysis, architecture, policy, and management functions: "if I'm really being cavalier, I say we're the same as building maintenance" (M-9). Because these government IT units focus on IT infrastructure and basic functionality, they have limited resources to apply towards OSS publication.

Participants viewed software as a means to an end, not an entity on its own: "we are not interested in OSS [consumption or publication] unless it provides value in building and delivering a product" (M-7). Similarly, participants in these units believed they should provide quality software, meet requirements, and provide software quickly. These participants believed the goal of developing software was to get it to work, to solve a problem, and not necessarily to share with others: "[the] goal is to provide value to our customers, if we don't deliver, the funds go away" (M-11).

Some participants believed publishing source code was not a priority because no other agency would have a use for it: “publishing the source code would not do anyone any good because what we do is very unique. That’s why we don’t do it” (M-22). Moreover, some participants saw code through a historical pre-Internet or mainframe code lens and as such did not believe the code would be relevant in 2019. The general consensus was the code was built for a specific reason, to solve a problem, and there was no reason to publish it.

Change in practices. This sub-category included data from participants who viewed change negatively, which hindered the unit’s frequency of publishing OSS. Beliefs about change in practice invoked fear, required the actual change itself, and required updated or new high-tech skills.

Change in practices included considering aspects of the change itself. Some participants did not want to be bothered to change to a new technology or new approach to current technology: “there is some amount of leave me the hell alone, this works great, I am not interested in learning that new crap” (M-5). Furthermore, with regards to the FSCP mandate for publishing OSS: “there are other people [in the unit] who think we should do the bare minimum to meet the mandate and not share at all” (I-1). Some participants in units that were publishing OSS faced governmental restrictions and organizational constraints from governance. The participants believed they should consume code as a step to begin building software and to allow coding on a public code platform, and some have moved in that direction, but not as many as they had hoped: “architecture review groups do not allow much open source use without justification. This causes a great deal of apprehension” (M-2). Another form of governance restriction often cited was work being done by contract: “publishing code is not a priority in some contracts. This hinders the process of pushing out code” (M-8).

Participants believed that some units have personnel with limited skills that hindered OSS publication. Some participants reported that their units were moving towards management and a configuration of technology, changing from doing "hands-on" software development to not doing any: "we had a handful of people who could do some pretty cool things with software but they've all left because there really wasn't anything to do" (M-9). This led to concerns about contractors knowing how to do more technically in comparison to their government counterparts. In some cases, contractors set the platform, set the technology direction, and federal employees might not have known enough to do this work or were unable to challenge what was being done: "a lot of contractors come in… [and have] a bias towards using a framework or latest set of packages…you have to keep using…the next thing [the latest software] even if the old thing would work perfectly fine" (M-14).

Consider risk. Responses from participants that clustered in this sub-category centered on the belief that publishing OSS was inherently risky. The beliefs about risks were related to transactions. These transactional concerns included a consideration of contracts and other aspects of building software, such as the process to publish code, deviation from an approved review process, and consumption of code along with trying new code platforms.

Some units did not code and focused on buying products and configuring software: "if you can't buy a solution for it, though, just continue doing it the way they're doing it or just not do it at all" (M-9). Similarly, participants believed vendors were responsible for the risk associated with software development and delivery, which ultimately gave vendors control of the code: "[the] contractor does not give us access to code" (M-7). However, some participants in other units were opposed to the vendor carrying all the risk for holding the code. These participants offered a rationale for why units chose to use contracts for software development:

"[it] is disconnected from the value of open source… either you are very knowledgeable in computer science or software engineering or those sorts of things, or you don't know anything about it and you don't want to deal with the headache" (F-2). Countering the approach to buying software, there were beliefs that units should not buy the same software product many times: "one of the things missing from the conversation is the risk of not publishing things and buying them ten times in a row which happens so often" (F-5).

Participants suggested establishing review processes to publish code in their units, but they believed there was a risk with these types of processes due to lengthy oversight reviews. If the code review process was daunting, developers would still publish code by avoiding review. Furthermore, locking down an environment forced developers to take alternate paths to publish code, which could cause additional risk to the agency if the code was not "clean." One unit developed a thirty-five-step code review process for all agency units to publish code: "some people will say everything that comes into the agency needs to be vetted, and you need to submit forms, and submit stuff, before it's approved by such-and-such board. Any developer worth their salt knows that's completely unrealistic to really implement" (M-6). Automated review tools were used, but these proved problematic as they could not capture all deficiencies in the code: "some say that everything needs to be vetted, everything that comes into the agency. But that is impossible to do because there aren't tools that can block everything" (I-2).

Consuming code and using code platforms were risky steps in publishing OSS. Developers wanted to consume code and believed security and code review processes were important: "we want to be able to continue consuming those but we want to be able to do it in a safe way. Security is paramount in everything we do" (M-5). There was resistance in some agencies to consuming open source, as evidenced by their architecture review processes that

limited units' ability to use non-COTS products: "if we come in and try to get something that's open source as a full product, it has a lot more trouble getting through the Enterprise Architecture approval" (M-2). To counter the negative perceptions of using and publishing OSS, some units tried public code platforms where units published finish code products or applications, but not the actual code: "anytime we push a release to production, we go back, grab a snapshot of that and then push that out to GitHub" (I-1). While this minimized risk for the unit, it did not result in publishing the application source code.

Need permission. Some unit participants believed permission was required to publish code. These participants believed they needed to obtain permission from their management and customers. Pertaining to management permission, one participant noted: "I certainly would not do it without approval from someone above me" (M-14). For units that were supporting other government agencies as customers, another participant suggested: "we would want to explicitly have permission to do that before we start" (M-11). In most cases, these unit participants pointed to policy as dictating permission because the policy articulated what a unit could and could not do with OSS: "OCIO has never put anything out recommending that we do that type of publishing" (M-2). Conversely, some units were publishing code and defaulted to open, which meant that all code should be published as OSS unless there is justification for not publishing in that manner. The participants in these units referenced policy as making OSS permissible and likewise, by documenting that policy in the public domain added clarity to their stance on publishing OSS: "I think a constant is the lack of clarity about what people can and cannot do [in government organizations with regards to open source]" (F-5).

Advantageous. Advantageous beliefs were based on perceived and realized outcomes. Responses from participants in this category were in units that published either frequently or

intermediately (44%). Even participants in units that minimally published often held advantageous beliefs about publishing OSS (36%) (see Table 5.2). Even though the variation was marginal, having advantageous beliefs played a role in publishing OSS. Participants with an advantageous orientation believed publishing OSS was important because it facilitated building a community, created efficiencies, demonstrated developer competency, and benefited the unit and citizens.

Table 5.2

Participants in Units with Advantageous Beliefs

Unit Publishing Frequency	Participants	% Sample
Minimally	9	36%
Intermediately	5	20%
Frequently	6	24%

Build community. Participants believed publishing code was beneficial because it created a community of sharing and collaboration: "together, a global community can create beyond the capabilities of any one individual" (M-6). Three main attributes were prevalent in participants' beliefs for building community: external contributions were beneficial, potential for reusability of code added efficiencies, and contributions allowed the unit to utilize code sharing platforms and practices for functions beyond publishing software.

Participants believed external contributions when publishing OSS provided value. Participation in public coding platforms made these external contributions possible. Units were responsive to code request updates, for instance, correcting issues and submitting PRs: "in the past couple [of years], we've had a couple of actual outside contributors helping us with our work, fixing a bug, or identifying problems. The more help we can get the better because we're never able to keep up with everything" (F-6). Participants described relationships and co-development with civil service, contractors, and academia. Agencies did not offer money; rather

they offered subject matter expertise where the end product was both beneficial to the public and the unit: “some code components and communities are much more advanced and the goal is to get all communities and code to the same level” (F-3). Additionally, some participants believed they should publish OSS to drive industry standards: “the reason we open sourced [our application] was mainly [to] build a community to influence industry best practices. Open source is the easiest way to get capabilities out to our customer sets with the least amount of bureaucracy” (F-4).

Participants believed in publishing OSS because of the potential reusability of the code and the ability to share knowledge and experience. Individuals external to the unit could use the code: “government paid for it which means it should be available to the public” (I-1). Sometimes the adoption path for code was not solely within the unit and in some cases state, local, and international governments could use the code. For state and local use, participants believed other governments could benefit: “knowing other agencies and states could benefit from using our product [allows for] collective knowledge around the product” (M-5). With regard to international adoption, a participant noted: “other countries were interested in what the U.S. was doing and they could just go and fork our repo and create their own site” (F-6).

Participating in a community with active contributions to a public coding platform provided a basis for interactions that extend beyond OSS, such as policy contributions. As one participant noted after working on a government-wide working group with the promise of using a code platform for collaboration: “I am actually really disappointed that they never did that. I think it would have been more efficient and effective to develop the document[s] in the open” (I-1). Further, documents including acquisition statements-of-work, policies, standard operating procedures, and guidelines, have been placed in a repository and used for public comment

allowing individuals to comment through PRs. This process invited feedback and peer review with full transparency of the process: "there is no reason the Federal Register could not be more GitHub like, allowing for comments in the open" (I-1).

Create efficiencies. Participants believed publishing OSS created efficiencies in the software development lifecycle because it allowed the unit to break down development silos, utilize new development technologies and processes (i.e., Development Operations or DevOps), and minimize redundant code.

Participants reported initiating software development in the open to take advantage of contributions from multi-unit teams: "we prefer to use open source software where possible. This allows for the code to be published immediately, blank repositories slowly form into something" (F-5). It also enabled the tracking of contributions and PRs. Having more OSS available facilitated the availability of tools to developers: "once this is recognized by those in the organization who may not completely understand or be on board, a different conversation occurs about what is possible" (M-4). According to some participants, publishing OSS created broader unit efficiencies: "we discovered that the easiest way to move software from one intelligence agency to another was to publish it on the Internet and let the other agency download it because that meant that we didn't have to create an interagency sharing agreement" (I-2). Finally, participants reported finding efficiencies with OSS because of the collective nature of working with this type of code: "it provides the ability to tap into the collective genius" (I-5).

Demonstrate competency. Participants believed publishing OSS demonstrated unit competency with software. This competency was the unit's ability to exercise authority to publish OSS. The unit justified publication based on professional ideals.

Unlike units that sought permission to publish OSS, some units exercised the authority to do so. This authority was tied to competency and the goals of the unit. One example was a unit that turned around a failed software development program: "we were given authority to do things different with software development…not typical of most programs in the agency" (M-5). Another example was a unit that published because the individuals in the unit demonstrated knowledge of the technology and articulated why it was important to share OSS: "we pushed for OSS because of understanding the difficulties of encryption and data compression technologies" (F-2).

Having the capability to publish OSS often aligned with participants' professional ethos. Some participants adhered to specific professional standards when developing and publishing software (e.g., Institute of Electrical and Electronics Engineers or IEEE, prescribed data science methodologies). Professional affiliations defined what should be done with coding products with specifications calling for visualizing, communicating, and sharing results: "we implement the data science discipline by developing data products with a strong preference to…open source technology" (F-2). With research units, individuals believed they should publish code as an artifact of their documented research: "we publish our software based on the scientific articles that get published because there's an increasing requirement to make that code available. And we have likewise increased our presence in publishing software" (F-3).

One way to begin using and publishing OSS was to test it on a periodic basis and then show the benefits to others. In some units, demonstration of capability occurred through incorporating a technology application. It has led to more open sourcing: "we did try a few things and then it worked, and more people bought into it. More people tried things and now we are at the point where we're really trying to start open sourcing more of our code that we're

making too" (I-1). Some units moved toward trying new approaches with concurrence of the CIO organization. One participant told a story of trying a new approach versus writing a justification and seeking approval prior to implementing: "we could have gone the traditional standards route and wrote a paper and try to push it through a standards organization like NIST or OASIS or something and waited 30 years. And at my retirement party, I would have found out if anything happened with it. Or we could have gone open source and actually try to influence stuff" (F-4). The participant chose the former approach, and his unit now runs a multi-agency open source platform.

Realize benefits. Participants believed publishing code provided benefits resulting in cost savings and transparency for citizens regarding what the government does with taxpayer funds as well as providing public access to the end product. Realizing benefits encompassed beliefs that OSS publication would help reduce technical debt, improve code quality, provide better documentation, provide cost savings, and help build a sustainable product.

Participants believed publishing OSS should result in cost savings. Participants often reported moving towards consuming and publishing OSS because of limited resources to purchase software: "[it's] a huge cost and time savings, to be able to reference those things and we sometimes are on the receiving end of those references, like we see something else we want to copy, but regardless of who it is, it is a net benefit for the world more broadly" (F-5). Another participant spoke more broadly about incorporating technology into the agency: "we don't introduce technology that doesn't save the agency money" (F-2). Finally, there was discussion of purchasing the same product multiple times by many agencies: "we shouldn't be doing the same thing one-hundred times, when we can just have an open source repo and then have a hundred people use it or one-hundred people fork it" (F-5).

Participants believed code should be shared because it provides transparency for citizens regarding one element of government spending as well as the public benefiting from source code developed by the government. Documenting code repositories and the code itself created long-term code viability, reduction of technical debt, and improved code quality: "we are using OSS to support maintainability, robustness, and extensibility" (F-3). Furthermore, participants believed vendors could also benefit from government published OSS: "vendors can reuse the government code and see business rules already baked-in then it ultimately helps the citizen in interacting with the government" (M-5). Similarly, vendors benefited from OSS: "releasing code also helps to create competition among vendors who take the code, make a product, and sell to citizens" (M-5). This competition potentially pushed for more robust features and lower costs. Finally, participants viewed themselves as public employees who developed products with taxpayer funds and thus should provide OSS to the public: "we believe that open source is a really good way to…demonstrate transparency as far as how effective and efficient steward we are of tax dollars" (M-13).

Both Cautionary and Advantageous. Forty-four percent of participants in the sample (11 of 25) expressed having both cautionary and advantageous beliefs. These participants were in units that published OSS less frequently (see Table 5.3). Although the individual participant may have held advantageous beliefs, the individual believed that the cautionary beliefs of the unit, and potentially the broader organization, prevented the unit from publishing more frequently. It is possible that other organizational factors in combination with beliefs need to be examined more closely, as individual beliefs alone did not allow the unit to overcome its more cautionary beliefs about scope, permission, risk, and change with publishing OSS.

Table 5.3

Participants in Units with Cautionary and Advantageous Beliefs

Unit Publishing Frequency	Participants	% Sample
Minimally	7	28%
Intermediately	2	8%
Frequently	2	8%

Summary. Cautionary and advantageous beliefs were associated with whether and how a unit publishes OSS. Cautionary beliefs centered on scope alignment, change in work practices, risk avoidance, and need for permission; advantageous beliefs included building a community around the software, creating development efficiencies, demonstrating competencies, and viewing code as being beneficial to the public.

Public Engagement

A second possible influence on unit frequency of publishing OSS was whether and how the unit publicly engaged with external units in or outside the agency. Public engagement involved collaborating with those outside the boundaries of the unit. The evidence can be placed in multiple theoretical categories and sub-categories (see Figure 5.2).

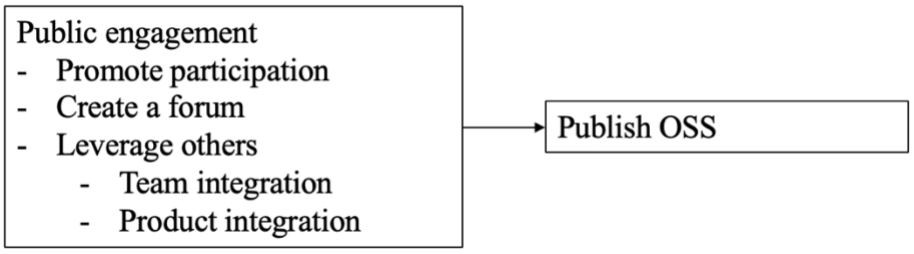


Figure 5.2. Public Engagement Categories and Sub-Categories.

Data indicated that participants in units that frequently and intermediately published OSS reported that their units conducted more public engagement (44%). Of the units that minimally published OSS, participants reported 40% conducted some public engagement, while 16% did not conduct any public engagement activities (see Table 5.4).

Table 5.4

Participants in Units Engaging Publicly

Unit Publishing Frequency	Participants	% Sample
Minimally	10	40%
Intermediately	4	16%
Frequently	7	28%

Note. Of the units that minimally published OSS, 16% did no public engagement.

Public engagement allowed the public to participate in problem solving and enhancing product quality. Participants reported using mechanisms to increase public engagement that resulted in forums for two-way communication (e.g., coding platforms, chat, webinars). Some participants reported integrating others and content (e.g., code, policies) into their units. From the data, I generated three themes of public engagement: promote participation, create a forum, and leverage others.

Promote participation. Participants valued and encouraged public participation and believed public engagement could be beneficial. Participants in units that intermediately and frequently published OSS (67%) reported their units also promoted participation (see Table 5.5).

Table 5.5

Participants in Units Promoting Participation

Agency Publishing Frequency	Participants	Percent
Minimally	4	33%
Intermediately	3	25%
Frequently	5	42%

Note. Percentages based on total number of coded responses.

Participants valued having input for products, code and policy, both from outside the unit and, potentially, from outside the agency as a whole: Because external collaboration does not occur, “I wouldn’t say poor products but whatever gets developed isn’t as good as it could have been” (M-6). Having more participation facilitated the engagement of external stakeholders,

which improved the understanding of their code and policy development needs. This engagement ultimately led to enhanced products that more effectively met stakeholders' requirements: "participation within and outside the unit could help us to update policies and processes" (F-6). Participation bred more participation: "it's weird, it creates this giant spider web. Like with the [university] case, some vendor knew us. It's not even a direct [connection to the open source platform], you get secondary, tertiary [connections]" (F-4).

Participants' units encouraged participation in multiple ways. One way was with contract language and deliverables: "[the] contract specified that their deliverables would be done as a pull request on GitHub" (I-3). Another way was with units promoting engagement through establishing a public presence on a code publishing platform to gather administrative and code contributions: "we would definitely consider it and include it. It's kind of the hope we've had" (I-1). Finally, respondents in multiple units offered they were in the process of transitioning from internal to community development: "we are in the process of moving the model core infrastructure that my group develops out to GitHub" (F-3).

Create a forum. Participants also reported utilizing mechanisms to solicit engagement from individuals outside their units. Forums for public engagement consisted of the unit communicating and collaborating with individuals outside their unit. The forums were in-person and virtual. These forums were used to place product (e.g., OSS code, applications, policies) and information online, to exchange product details, and to accept product changes in real-time. Participants came from roughly the same number of units creating forums that were minimally publishing as were those intermediately and frequently publishing OSS (see Table 5.6). Two ways in which units created a forum was with electronic tools and in-person engagements.

Table 5.6

Participants in Units Creating Forums

Agency Publishing Frequency	Participants	Percent
Minimally	7	54%
Intermediately	3	23%
Frequently	3	23%

Note. Percentages based on total number of coded responses.

Participants reported using electronic tools with individuals outside their units and in rare cases with the public. The most common tools for collaboration on OSS publishing were code platforms (e.g., GitHub), chat (e.g., Slack), webinars, websites, developer portals, and virtual meetings (e.g., Zoom). Some tools only allowed for one-way communication, while others allowed for two-way communication that added flexibility for more collaboration. The tools were typically used for OSS development and publication; however, some units used the tools for more: "we force GitHub use for things it's not designed for, like open data, file share, and documentation of sites" (M-4). Participants in units publishing OSS often used code publishing platforms because the platforms allowed for contributions, code changes through PRs, copies of code, and reuse. Public engagement was pursued through these code platforms with built-in communication mechanisms.

Participants also reported collaborating through in-person engagements. In some instances, these would have a virtual component via electronic tools. Examples of in-person engagements included Tech Talks, conferences, coding events, and inter-agency workgroups. Two units from different agencies held Tech Talks once a month where they invited guest speakers to discuss a technology topic, and it was open to all government agencies. A workgroup was established with the implementation of the FSCP brought agency members together biweekly to discuss the policy and its implementation progress at each agency. Units participated in hackathons where they invited public participants to come into the office to work

on unit software development and to compete for prizes. This was a new approach to engagement: "[it] was a paradigm shift for the agency because it required agency people to work with citizens in problem solving" (I-5).

Leverage others. A majority of participants (61%) reported integrating outside team members and open source products in units that published OSS more frequently and intermediately; moreover, five participants in units that minimally published OSS also reported their units sought to leverage others (see Table 5.7). Two main themes were team integration and product integration.

Table 5.7

Participants in Units Leveraging Others

Agency Publishing Frequency	Participants	Percent
Minimally	5	38%
Intermediately	3	23%
Frequently	5	38%

Note. Percentages based on total number of coded responses.

Team integration. Participants reported seeking help from others in the form of consulting and "hands-on" engineering support leading to OSS use and publication. This theme split into units enlisting the executive branch's two major digital service teams, the GSA's 18F group and the Executive Office of the President's (EOP) U.S. Digital Service (USDS). 18F is a fee-for-service program made up of federal employees who agencies can be hired to work on engineering projects. USDS is similar to 18F, but it only works on the government's largest technical challenges (e.g., Healthcare.gov in 2013) as mandated by the EOP. Both digital service teams default to an "open first" stance with OSS, and most units that work with them follow suit. Participants engaged with groups like 18F for input into acquisition and policy: "18F came in and helped us create contract mechanisms that would allow an Agile approach" (M-5). In other cases, these digital service teams began a project for the White House or an agency and then later

transitioned the software product to the agency itself, initially with support: “after transitioning the product from 18F, we had them work with us to help make it sustainable in their model of development and open sourcing” (F-6). Another example in which a unit engaged with an external group for support was related to an OSS project where the unit created a design standard for its product by working with USDS: “our designers have also been working with the USDS designers and the general design community across government on issues” (M-4).

Some units integrated teams and published OSS with other government units, contractors, and academia. One unit sought to develop an OSS practice to release more software and was directed by their management to research if anyone had already done what they were attempting: “one thing our director wanted us to do was go out and find out if there’s any other organization which already has a process we can leverage. So, we tried in terms of collaborating and formed a task force and developed this process as a group” (I-3). Participants spoke of using grants to collaborate with academia where all deliverables were required to be open source. Finally, participants also reported engaging with others for software development: “we have a very active open source community engaging in co-development with [agency] civil service and contractors” (I-5).

Product integration. Many participants sought out solutions to incorporate into their units and to contribute back to the OSS community. This occurred by copying others’ work and modifying it to fit unit needs: “the closest we’ve come to that [using others’ policy to create our own] is forking the Digital Service Playbook…we were experimenting with doing something similar to that for the community” (F-1). Similarly, with regards to software development processes and OSS publishing, another participant noted: “[we] forked GSA’s processes for open source software creation and publication. That was a starting point for the agency and unit

specifically" (F-3). Beyond copying and modifying the software, some units worked across government on a product and then brought it in-house: "we transitioned our product away from being managed, 18F developed it and designed and ran it for like three or four years before we actually transitioned it to be managed in house" (F-6).

Participants reported cooperation moving beyond OSS by seeking solutions for strategy, acquisition, and policy creation. With regard to strategy creation based on a copied informational repository, a participant noted: "[it's] really open data and data strategy, it's really been…GSA" (I-5). Similarly, units have been copying and using the content products as templates for acquisition and policy: "we created our policy by forking another agency's code policy" (I-4). However, it has not been an easy process for policy creation for those who are not familiar with code sharing platforms and electronic tools for public engagement: "even from our own team, people who engage in the policy process are not so hands-on with collaborative tools" (M-4).

Summary. Public engagement and its associated mechanisms were associated with whether and how often units publish OSS. Units publishing more frequently did so to promote participation, create forums, and leverage others. This was accomplished with electronic tools and bi-directional communication. Units that published more frequently often engaged with individuals outside the unit and worked with team members from digital service organizations.

Structural Dimensions

A third set of variables examined for possible relationships with whether and how a unit published OSS were the structural dimensions of the unit **(**centralization, formalization, differentiation, and coordination) (See Figure 5.3).

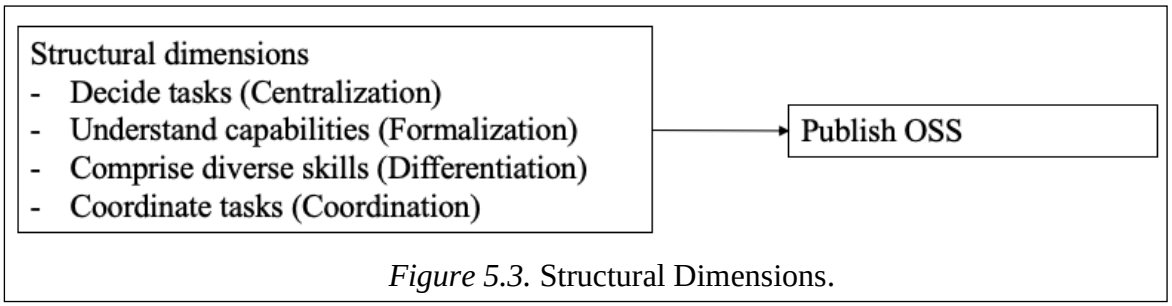


Figure 5.3. Structural Dimensions.

The evidence supported generation of four categories – decide tasks (i.e., centralization), understand capabilities (i.e., formalization), comprise diverse skills (i.e., differentiation), and coordinate tasks (i.e., coordination).

Decide tasks. According to participants, decision-making in their units occurred in either a directed or a participatory manner. Directed units accounted for 38% and participatory units totaled 62% of coded responses (see Table 5.8).

Table 5.8

Participants Reporting Unit Decision-Making Type

Decision-Making Type	Participants	Percent	Frequency of OSS
Directive	5	38%	F = 0, I = 0, M = 5
Participatory	8	62%	F = 4, I = 3, M = 1

Note. Percentages based on total number of coded responses. OSS publishing frequency: F = Frequently, I = Intermediately, M = Minimally.

Directed decision-making meant that guidance on task selection and prioritization came from upper management, customers outside the unit, and contractors or through specifications set in the contract itself. Regarding task priorities assigned by upper management, one participant responded: “we setup our priorities based on service demands from our customers. [The priorities] are affected by funding and by management. [Management] decide[s] that we need to go in this direction” (M-7). Another participant responded similarly regarding product management and guidance from upper management: “a lot of decisions still come from senior leadership when the product owner should have full control of the product” (M-4). Participants

reported that units relied on customers or stakeholders outside the unit to set direction and task priorities: “unfortunately the priorities are coming from a different group with what is due at this time” (M-12). Some units depended solely on contractors or requirements specified in a contract: “we have a fixed-price contract and we have what we call a list of releases” (M-2). Participants from units using directed decision-making used minimalistic project management tools to assist with task prioritization. One unit used a markdown file to track tasks: “if someone comes up with a new enhancement or something or bug fix…we just go directly to the repository and go to our backlog markdown file” (M-6). In other cases, federal employees used no task management tools and relied on a contractor: “our contractors use JIRA and Trello but we rarely get to see that. They give us an update on the backlog” (M-7). Notably, only participants in units that minimally published OSS reported more directive decision-making.

In contrast, participants in units more frequently publishing OSS reported using participatory decision-making along with collaboration, supported by electronic tools, to prioritize tasks to determine work to be performed. Some participants said upper management did not dictate direction of the unit and units had autonomy to explore their own priorities: “so the short answer is we are very hierarchy [sic] of things to get done in the fiscal year but we have some freedom to explore or present new ideas, depending on what needs to get done” (F-2). In some cases, task priority was more collaborative: “there isn’t really a structure of telling people what to do” (F-5). Another participant’s response was more extreme: “my contractors told me that we need a little more structure” (F-4). Participants reported the desire to include others in decision-making: “we work hard to make sure we are not leaving anyone out” (M-5). Finally, for some participants, it was all about progress and deliverables: “as long as we’re getting stuff done that benefits other people, we’re removing barriers and we’re demonstrating forward

progress" (I-2). For electronic tools to manage projects and tasks, units demonstrating this type of decision-making spoke mostly about Agile methodologies and tools to complement that type of development: "prioritization is a huge part of everything we do. We are Agile and our users are heavily engaged. Not only in identifying what needs to be developed but in the prioritization of how it gets developed" (M-5). Other units decided on tasks and tools: "each team decides how to work on tasks and what technical tools to use (e.g., Trello, GH issues)" (F-5).

Understand capabilities. Some participants reported that units using documented procedures and policies (64%) resulted in lower frequency of OSS publication (see Table 5.9). Participants across units, regardless of publishing frequency, expressed the need for more formalization.

Table 5.9

Participants Reporting Understand Capabilities

Type of Procedures	Participants	Percent	Frequency of OSS
Documented	9	64%	F = 2, I = 2, M = 7
Undocumented	5	36%	F = 0, I = 1, M = 4

Note. Percentages based on total number of coded responses. OSS publishing frequency: F = Frequently, I = Intermediately, M = Minimally.

Participants in units that intermediately and frequently published OSS reported relying on guides and documentation, created either by the unit (e.g., developer guidance, organizational manuals) or by professional organizations (e.g., IEEE specifications, data science frameworks) for creating and publishing OSS. In some cases, professional guidance (i.e., IEEE) superseded organizational guidance, allowing OSS publication to occur more readily. With reference to releasing OSS and using professional standards, one participant noted: "we modeled a process after an IEEE engineering task force where diverse organizations with sort of competing standards and priorities [can publish software]" (I-4). Some participants reported relying on policy documentation offering instructions on how to conduct operations: "Policy. It is a huge

thing for us. Policy compliance is embedded in everything we do, we examine everything we do, all has to be very policy compliant" (M-5). External policies also pushed units to publish more code, as multiple participants suggested. One example was "we have to meet the mandate [FSCP] that's coming out of the White House" (M-1). Forcing units and agencies to document guidelines and policies helped them to better understand their operational boundaries: "that's a big decision to make [to publish OSS], and if you force someone to write that down or publicly write that down, that forces people to overcome…hurdles" (F-5).

Participants in units that minimally published OSS provided a rationale for the lack of documented guidance and policy: "OCIO has never put anything out recommending that we do that type of publishing. So, we would be doing it on our own and it would not be with the limited resources we have" (M-2). Other units in the same situation, having a desire to publish OSS but lacking formal documented guidance, echoed this sentiment: "We want to contribute a lot more but our hands are tied in some aspects because we do not have an open source policy" (M-4). In one case, there was uncertainty about whether documentation existed: "the processes that we adhere to [with regards to pushing software through the development stack] may be on our agencies intranet" (M-3).

Comprised of diverse skills. Units with individuals with more diverse skills more frequently published OSS (see Table 5.10).

Table 5.10

Participants Reporting Unit Skill Diversity

Type of Skills	Participants	Percent	Frequency of OSS
Diverse Skills	7	50%	F = 4, I = 2, M = 1
Non-Diverse Skills	7	50%	F = 0, I = 1, M = 6

Note. Percentages based on total number of coded responses. OSS publishing frequency: F = Frequently, I = Intermediately, M = Minimally.

Participants in units with more diverse skills could have individuals in multiple roles: "we were lucky to end up with people who coded at one point in their life" (F-2), and "we hire people with strong depth of knowledge in one area and then tangential knowledge in a wide area of subjects" (F-5). Integration of teams consolidated diverse skill sets, so one unit could work across multiple business lines: the unit not only included software developers but also included an engineering team, acquisition team, and policy team.

Participants reporting less diverse skills were in units that published less OSS. Federal project and program managers in 2019 focused more on the management of a product or project, often by overseeing a contract: "historically, the organization has changed in the past 15yrs…the people in charge said it was more effective to contract out the actual technical work and make sure there are good feds supervising it" (I-1). This has made units less technical and less able to oversee the work: "as an organization moves away from…software development, it becomes less technical. As it becomes less technical, it becomes less able to oversee and project manage and buy software development" (M-9). In some cases, contracts did not allow for diversifying skills: "they [the contracts] are specific to the need of the time because most of them are contractors" (M-10).

Coordinate tasks. Participants who collaborated and communicated were in units that published OSS more frequently. There was less coordination overall in the sample, but the frequency of OSS publication increased with more coordination (see Table 5.11).

Table 5.11

Participants Reporting Coordinate Tasks

Amount of Coordination	Participants	Percent	Frequency of OSS
More Coordination	8	44%	F = 3, I = 1, M = 3
Less Coordination	10	56%	F = 1, I = 3, M = 6

Note. Percentages based on total number of coded responses. OSS publishing frequency: F = Frequently, I = Intermediately, M = Minimally.

Participants reported collaborating and communicating more informally and frequently as a way to share information and solicit support when working on projects with tight deliverable deadlines: “we are big into share. We spend an enormous amount [of effort], we’ve got incredibly tight timelines and goals. We do everything we can to bring others on board” (M-5). Moreover, units were collaborative when working on engineering tasks: “it’s some serious collaboration like everybody knows, you’ve got a voice, speak up. If you know a better way of going about it or you have a thought about it, then say something” (M-1). Other participants in research units reported using both formal and informal communication mechanisms; they used formal agreements with universities to scope work, but informal mechanisms for collaboration. Common methods for coordinating tasks included brainstorming, huddles, and stand-ups. One unit running a large open source project relied on outside coordination with participants in the military and in other agencies: “our open source project thrives off of informal communication and collaboration” (F-4). Finally, electronic tools were used to facilitate communication and collaboration: “all of our collaboration has to be done using the tools like Google Hangouts, Slack, or that sort of thing. There is frequent amount of engagement” (F-6).

Participants who reported less coordination and less frequent use of electronic tools were in units that tended to publish less OSS. In some cases, the contractor managed the work and coordinated with federal employees on progress: “our contractors use Trello and JIRA for development. We rarely get to see that. They tell us, okay next sprint, this is what we are working on.” Other participants reported experiencing work in relative isolation where they did not collaborate with one another: “we are extremely siloed, [others have tried to] create their own org charts” (M-6). This led to limited support from other units: “we have a problem that other parts of the [agency] or not acting on our actions list” (I-2). Finally, participants reported less

collaboration when the tools were more application-based (e.g., ticketing) or when they used email: “we don’t have Slack or anything like that, we literally run everything on Outlook. And so, what happens, you have an email system as the way people communicate, they write office documents, we have this extra layer of office documents on top of what they could have written in plain email” (F-2).

Summary. Variations in four structural dimensions – centralization, formalization, differentiation, and coordination – were associated with whether and how frequently units published OSS. Units publishing more frequently relied on less centralization with participatory decision-making, more formalization with documentation, more differentiation by having diverse skills within the unit, and more coordination with collaboration and use of electronic tools.

Organizational Location

A fourth potential influence on OSS publication was a unit’s organizational location. Organizational location was a less impactful structural dimension in the technology-structure literature, but it plays an integral part in many organization theories. Here, it was associated with units’ frequency of publishing OSS (see Figure 5.4).

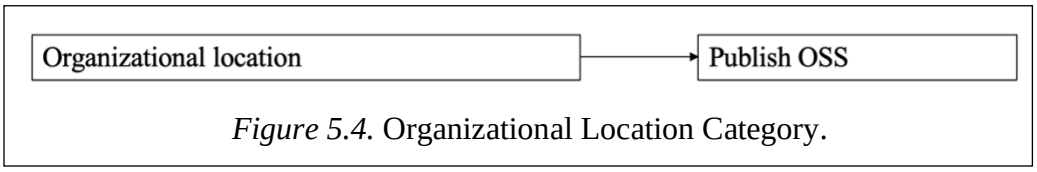


Figure 5.4. Organizational Location Category.

Sixty-eight percent of participants in this study were in units located outside of the agency CIO office, and many of these units published OSS more frequently (see Table 5.12). Even so, eight participants in units located outside the agency’s CIO office reported their units only minimally published OSS.

Table 5.12

Participants Reporting Unit Location

Type of Office	Participants	% Sample	Frequency of OSS
Non-CIO Office	17	68%	F = 6, I = 3, M = 8
Sub-CIO Office	3	12%	F = 0, I = 0, M = 3
CIO Office	5	20%	F = 0, I = 2, M = 3

Note. OSS publishing frequency: F = Frequently, I = Intermediately, M = Minimally.

OSS publication mostly occurred in units with fewer organizational reporting layers (i.e., less hierarchy), as multiple layers potentially added bureaucracy, limited authority, and required approvals. Single-layer units consisted of an executive or individual with executive trust: "we are in a unique position because of being close to the executive. We have a lot of competing priorities but this one has been delegated" (I-3).

Participants identified their units' location by conceptualizing their organization as a metaphorical circle, as depicted in Figure 5.5. The center of the circle represented the functional or headquarters unit that was interested mostly with internal affairs (e.g., budget, strategy, technology deployment). The middle of the circle included units that interacted with both the center and the boundary of the organization with a focus on functional and on customer-facing tasks. Units at the boundary of the circle focused on a product or customers outside the organization.

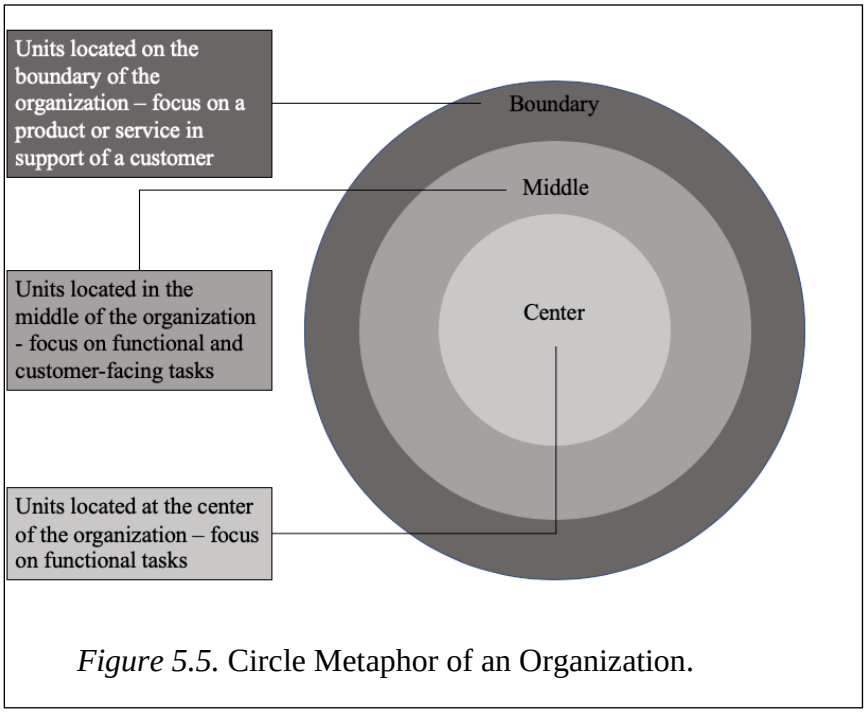


Figure 5.5. Circle Metaphor of an Organization.

Participants were evenly distributed in units in the center of the organization (32%) and those in units at the boundary (32%). Slightly more participants' units (36%) were located in the middle of the organization, and those had higher publishing frequency (see Table 5.13). Though the data had marginal numerical differences, participants open-ended responses indicated that their location was more often associated with the middle of the organization.

Table 5.13

Participants Reporting Unit Location vs. OSS Publishing Frequency

Type	Participants	% Sample	Frequency of OSS
Center	8	32%	F = 1, I = 2, M = 5
Middle	9	36%	F = 3, I = 5, M = 1
Boundary	8	32%	F = 2, I = 2, M = 4

Note. OSS publishing frequency: F = Frequently, I = Intermediately, M = Minimally.

Units in the middle of the organization were able to try more technology approaches and tools. As one participant reported when his/her agency's CIO office was interested in the results of testing new technology: "we had support from the CIO office leadership. They didn't tell us

to do it but they said they would be supportive…and they said, 'we would love to see how it goes'" (F-1).

Summary. Organizational location was associated with whether and how frequently units published OSS. Units located outside the agency CIO office (metaphorically located in the middle of the organization) had fewer layers of hierarchy and tended to publish OSS more frequently.

Summary of Results

Overall, the findings suggested that cultural beliefs, public engagement, structural dimensions, and organizational location were associated with units' frequency of publishing OSS and was helpful in highlighting the degree to which units implemented FSCP. Cultural beliefs consisted of cautionary and advantageous categories. Units with cautionary beliefs minimally published OSS. Participants in such units believed that publishing OSS was not within the scope for how they normally built software, was negatively associated with change, created risk for those who would publish and for the agency writ large, and required permission either in the form of guidance or through verbal confirmation of a supervisor. Participants in units with more cautionary beliefs were consistent with a monolithic agency culture where publishing OSS was not required, valued, permitted, and ultimately not the norm of software development.

Those with advantageous beliefs appeared in units across all publishing frequency categories. Those in units with advantageous beliefs thought that publishing OSS was good for building a community, creating software development efficiencies, providing demonstration of competencies, and being beneficial to the unit and the public. These units supported sub-cultures by recognizing the competency of individuals and by granting authority over software development.

Some participants, in units that less frequently published OSS, expressed both cautionary and advantageous cultural beliefs. This occurred when individuals believed using and publishing OSS was advantageous, but encountered cautionary beliefs either within the unit as a whole or within the broader organization, which in their view prevented the unit from publishing OSS more frequently.

Public engagement included promoting participation, creating a forum, and leveraging others with team and product integration within a unit. Units that published more frequently employed more public engagement mechanisms with bi-directional communication. Public engagement could have been with external units in the agency, across federal agencies, or with the public. A unit's starting point for public engagement was with a code publishing platform but extended further to public forums, events, and online tools. In a few cases, integration occurred when units would incorporate code and documents (e.g., policy, acquisition) and team members from new digital service organizations.

The four structural dimensions were associated with units' frequency of publishing OSS. Centralization consisted of directed and participatory decision-making. Directed decision-making units less frequently published OSS and these units made decisions in a more hierarchical fashion where a manager or contractor determined tasks to be performed with limited input from the software development or program management team. Participatory decision-making units more frequently published OSS, and members of the unit and sometimes others outside the unit made decisions in a collaborative manner. This was often magnified with electronic collaboration (e.g., chat) and project management tools (e.g., Trello).

All units, regardless of publishing frequency, valued understanding capabilities or formalization. Units publishing more frequently often documented and published their

procedures, usually on code sharing platforms while publicly promoting their position on the topic. Participants in units publishing less frequently cited lack of documentation as a reason not to publish OSS.

Having diverse skills or differentiation meant that units consisted of individuals with diverse skill sets. Units with individuals who had one skill published less frequently than units with individuals with diverse skills. Units with diverse skill sets had individuals who could accomplish multiple technical tasks (e.g., gather data, write code to analyze the data, code a webpage to display the data).

Coordination related to how units collaborate. Units that more frequently published OSS collaborated more and did so informally, which contrasted with units that published less OSS. Units publishing more frequently also used electronic collaboration tools (e.g., chat, virtual meetings).

Organizational location was a unit's location within the organization. Participants identified where they perceived their unit resided within the organization based on a circle metaphor. Most units in the study were located outside of the CIO's office, which was responsible for implementing the FSCP, and typically had the most software development resources. Based on open-ended responses, units located towards the middle of the organization more frequently published OSS. This is different than in the past where OSS in government mostly occurred at the boundary of the organization (in, e.g., labs, space centers).

The concluding chapter discusses the implications of these findings for policy, public administration, and organization theory scholarship.

Chapter 6 Summary and Conclusions

This chapter summarizes and discusses this study's contributions to understanding implementation of the mandate to publish OSS. It concludes with implications, limitations, and recommendations for future research.

Overlapping Variables Affecting OSS Publication

Among the key findings was evidence of relationships among advantageous cultural beliefs, public engagement, participatory decision making, and diverse employee skills.

Cultural Beliefs. The findings from this study aligned with Schein's (2010) notions about how culture affects organizational behavior: beliefs supported how units made predictions and selected actions, and they helped explain observed and enacted behaviors when manifested in particular situations. A unit's cultural beliefs were based on participants reports on members' perceived notions and actions with the technology. Units publishing more frequently often exhibited what Martin and Siehl (1983) called orthogonal subcultures, with individuals who accepted the core values of the parent or larger agency culture while maintaining separate, nonconflicting values of the unit. Units with cultures characterized as having "advantageous beliefs" published OSS more frequently than units with cultures featuring more cautionary beliefs. Some participants in units that less frequently published OSS expressed both cautionary and advantageous beliefs. This suggests that beliefs alone do not determine whether a unit could publish OSS more frequently; the advantageous beliefs of some members did not help the unit overcome perceived restrictions (i.e., scope, risk, change, permission) in the unit or in the broader organization.

Public Engagement. Public engagement was an action-oriented approach involving individuals from outside the unit. Public engagement has been associated with effective

programs by allowing individuals to come together within a scientific and technical domain to ask questions and solve particular problems (Feldman & Khademian, 2007). Participants in most units included here regardless of publishing frequency reported participating in public engagement activities. However, units that published OSS more frequently conducted more and varied public engagement activities including at least two of the three public engagement sub-variables (promote participation, create forums, and leverage others). Furthermore, units that conducted public engagement published OSS more frequently by using bi-directional information flows, a variety of in-person and virtual events, and multiple electronic tools. Units publishing less frequently tended to conduct only one public engagement activity and did so less often.

Participatory Decision Making and Diverse Skills. Participatory decision making and diverse skills were values of structural dimension that were associated with cultural beliefs and public engagement. Potentially they could offer insight into why some units that publish OSS more frequently were able to overcome cautionary cultural beliefs and limited public engagement activities. In order to publish OSS more frequently, units evidently needed to exercise advantageous cultural beliefs, be involved in more and more varied public engagement, encompass participatory decision making, and include those with diverse skills (see Figure 6.0). Further research should be conducted to determine the strength of the relationships between and among these variables.

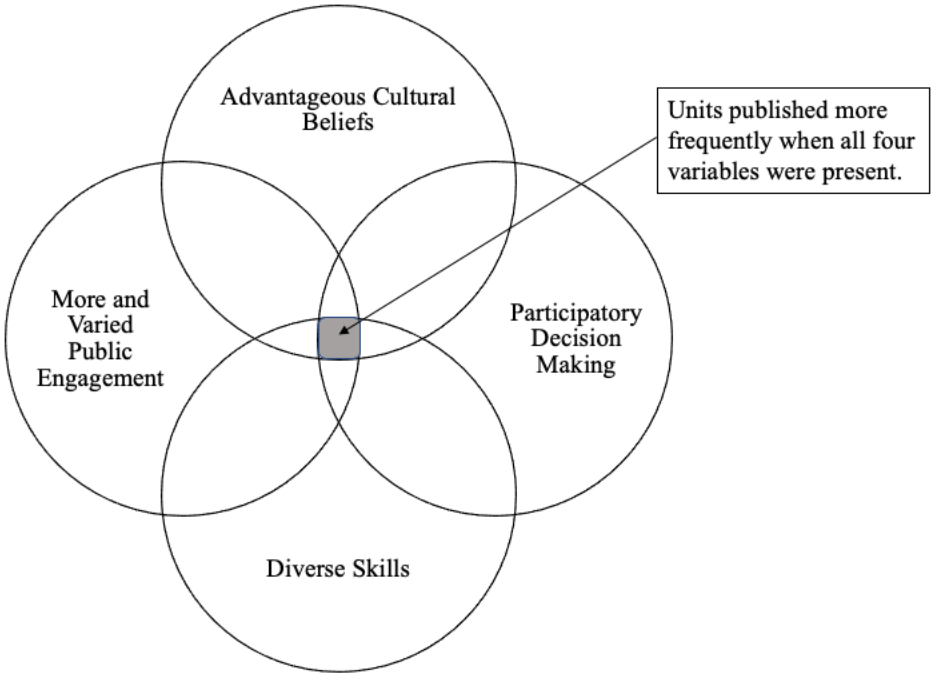


Figure 6.0. Relationship Among Variables Affecting OSS Publishing Frequency.

Structural Dimensions Affecting OSS Publishing

The technology-structure literature provided insight into whether and how structural dimensions were associated with units' frequency of publishing OSS. Related dimensions included organizational centralization, formalization, differentiation, and coordination. (See Figure 6.1).

An example of an environmental disturbance was the new technology policy, the FSCP, which all software units in the 24 CFO Act agencies were mandated to follow. Units that exhibited less centralization, more formalization, more differentiation, and more coordination published OSS more frequently, and thus complied with the mandate. In this study, the findings on formalization were contrary to those in much of the technology-structure literature, which reported that less formalization resulted from environmental uncertainty. All the participants in

this study thought it was imperative to have written policies in order to publish OSS, even beyond what the FSCP stated, suggesting that more formalization may be necessary.

This study examined these dimensions qualitatively, so they were not measured distinctly. Therefore, the qualifiers related to each dimension (less or more) were more subjective. More holistically, though, those units that were less centralized, more formalized, more differentiated, and more coordinated appeared to be more likely to publish OSS.

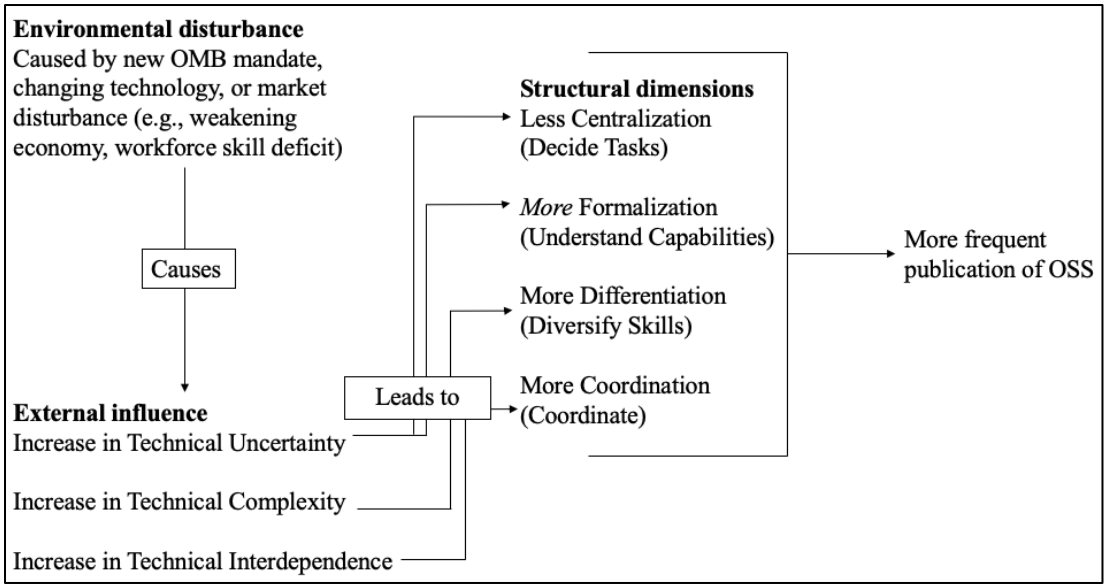


Figure 6.1. Technology-Structure Affecting Unit Frequency of Publishing OSS. Emphasis added to formalization based on data collected in the study demonstrating the opposite of what was previously found in the technology-structure literature.

Diverse Skills. Skill diversity occurred because of greater technical sophistication (Sorge, 1989). Units with participants with diverse skills published OSS more frequently. The units that participants identified as having diverse skills included federal employees or had agreements with other federal organizations to employ external federal employees through inter-agency agreements (e.g., arrangements with GSA's 18F). Instead of units purchasing one type of labor skill with a contracted employee, the units brought in federal employees with multiple

skills (in, e.g., data science, software development), which evidently resulted in publishing OSS more frequently. The use of federal employees with more diverse technical skills provided evidence that may counter the alleged effects of the Hollow State.[25] Arguments about the emergence of a "Hollow State" appeared as governments used third parties, often non-profit and private firms, to deliver taxpayer-funded goods and services. This reliance on third party services in this view has resulted in decreased government performance and loss of legitimacy of government action due to limited monitoring and control by government agencies (Milward & Provan, 2000).

Organizational Location and OSS Publishing

Where a unit resided in the broader organization influenced the frequency of its publishing OSS. More units outside the CIO office published OSS. The units publishing OSS were metaphorically located in the middle of the organization where they could interact with the CIO to understand the FSCP, while also focusing on product delivery and customer support (see Figure 6.2). Furthermore, OSS publishing units located in the middle of the broader organization had fewer layers while being close to executives who provided authority to publish OSS. This counters previous findings that units that published OSS were located at the boundary of the organization (in, e.g., labs, space centers).

[25] Code created by contractors that is paid for by the federal government is often contractually the property of the contractor. Code developed by federal employees is the property of the government. Another aspect of the FSCP is for agencies to update acquisition language to capture new code developed by a contractor for potential release as OSS.

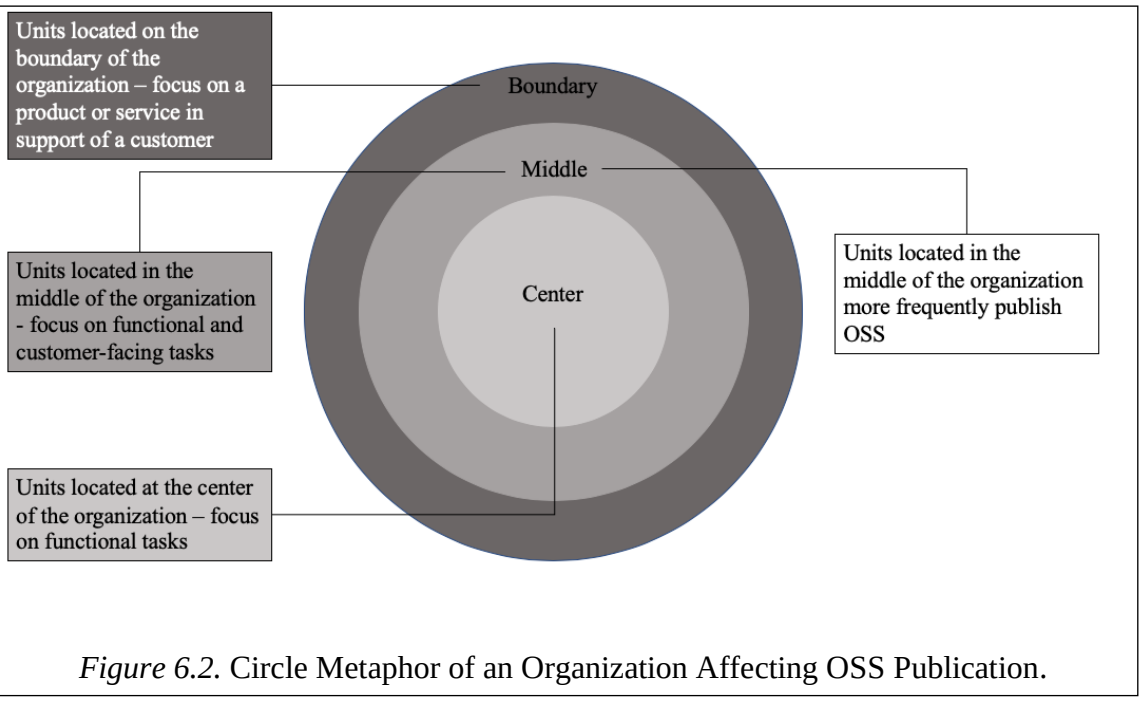


Figure 6.2. Circle Metaphor of an Organization Affecting OSS Publication.

Additional Analyses for Theory Building

Researchers (Gregor, 2006; Holton & Walsh, 2016) have described theory as descriptive and explanatory and as enhancing an understanding of the world allowing for predictions of what might happen in the future. Additional analyses like the one here provides detail about the key variables in developing a theory of implementing a directive such as the OSS publication mandate (see Figure 6.3). The agencies and corresponding units included in this sample exhibited actions describing and explaining an environment either that supported the implementation of the FSCP for the publication of OSS or that did not. Attributes of the variables for supporting OSS publication included cultural beliefs that accepted the perceived benefits of OSS while also encouraging experimentation with OSS publication. Participants in units were more likely to publish OSS if that publication aligned with occupational and professional values. Public engagement with bidirectional information flows through in-person and virtual events and mechanisms provided an environment supporting OSS publication. Warding off technical uncertainty, complexity, and interdependence was possible with less

centralization, more formalization, more differentiation, and increased coordination. Units with an organizational location outside the CIO office, metaphorically in the middle of the organization with fewer hierarchical layers, were more likely to implement the policy for publishing OSS.

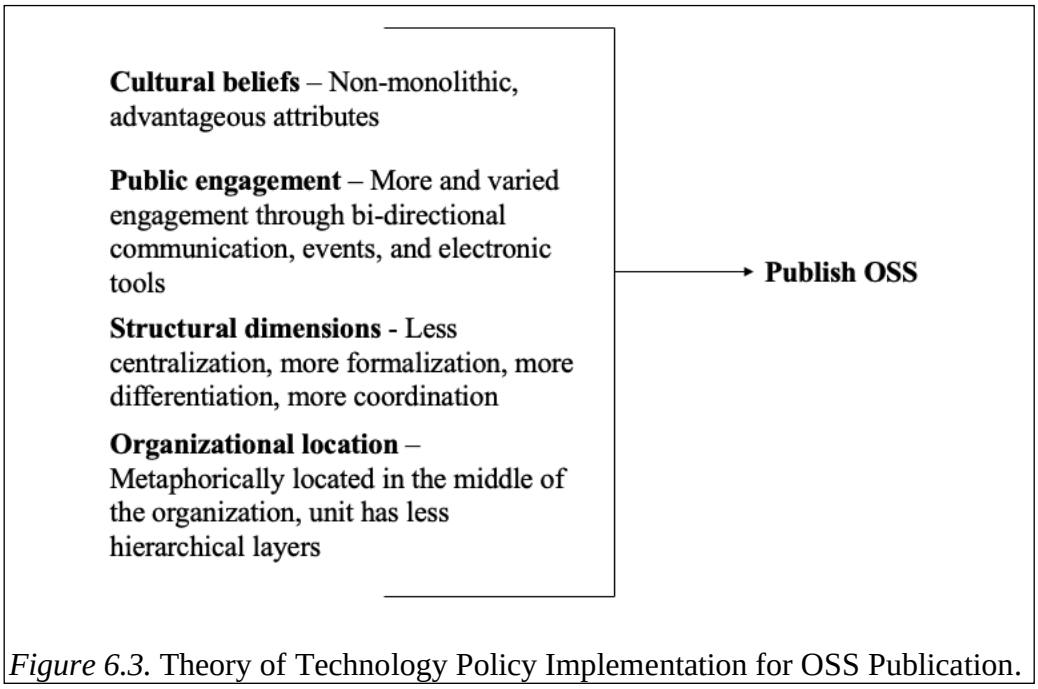


Figure 6.3. Theory of Technology Policy Implementation for OSS Publication.

Implications of the Study

The findings from this study have implications for policy, public administration, and organization theory. These findings inform public administrators and public administration scholars about an approach to enhance implementation of technology policy.

Policy. Policy is comprised of the goals public organizations wish to achieve and provides the means required to do so. "Implementation is the carrying out of a basic policy decision, usually made in statute (although also through important executive orders or court decisions). Ideally, that decision [to implement a policy] identifies the problem(s) to be addressed, stipulates the objective(s) to be pursued, and, in a variety of ways, 'structures' the

implementation process" (Sabatier & Mazmanian, 2005, p. 540). The study's findings have several implications for policy implementation.

Units require explicit policy to implement technology. Policy publicly expresses federal government priorities. Beginning in the Obama administration, federal policy focused on open government initiatives, outlining how to make government and government-produced artifacts more accessible to the public ("Open Government Initiative", n.d.). The government uses websites and dashboards to share technology artifacts and policy implementation progress. For example, in 2013, USCIO published the M-13-13: Open Data Policy – Managing Data as an Asset and directed agencies to account for data as a digital asset, to inventory the data, and to release datasets publicly. M-16-21: Federal Source Code Policy: Achieving Efficiency, Transparency, and Innovation through Reusable and Open Source Software followed in 2016 and directed agencies to inventory source code and release a portion as OSS. This study found units required formal policy to release software as those who did not publish frequently said they could not because there was no guidance to do so. Respondents in units that frequently published OSS said they were allowed to do so because their internal policy complemented the FSCP. To support the implementation of policy, public administrators should work in concert with software development units to articulate an internal policy that aligns with the FSCP and that outlines what the units should do regarding OSS publication.

Even with policy, it is often unclear what units are operationally required and allowed to do. "The policy process requires a knowledge of the goals and perceptions of hundreds of actors throughout the country involving possibly very technical, scientific, and legal issues over periods of a decade or more" (Sabatier, 2007, p. 4). This pulling together of many stakeholders resulted in a policy that typically was too vague for agency software units to implement effectively.

Consequently, units implemented policy how they deemed appropriate leading to myriad results, both positive and negative. Further, units lacked clarity about whether code could be released and about how much interaction the units could or should have with the public. When public administrators engage with their development units, transparent and explicit expectations should be outlined in documented policy, so that units understand the boundaries of their work in publishing OSS.

Policy contributes to competing interests among units. Numerous factors affect policy implementation including media attention to the problem, public support, socio-economic conditions, commitment and skill of leadership, technology, and financial resources (Sabatier & Mazmanian, 2005). Each of these factors had an effect on units as they competed for limited resources in an attempt to implement numerous executive policies, laws, and internal operational mandates, in addition to the FSCP. These policies could last indefinitely and could even conflict with other policies and laws already enacted within the organization. In most cases, agencies attempted to comply with policy without the accompanying financial or technical resources from the policy provider. Therefore, when solidifying internal policy around publishing OSS, agencies should allocate appropriate technical and financial resources and try to defuse possible competing interests.

Public Administration. Public administration is a broad field of scholarship and practice encompassing aspects of administering policy and managing programs to produce public goods (Brudney, O'Toole, & Rainey, 2000). The findings of this study have implications for public administrators pertaining to culture and public engagement.

Public administrators should nurture advantageous cultural beliefs. Public administrators should seek opportunities to create shared experiences, enhance professional skills

related to software development, develop risk and change management plans, and align scope of work to incorporate software development with OSS. "Culture arises through shared experiences of success" (Schein, 2010, p. 55). Units that evaluated and integrated OSS software practices into their development lifecycle gained legitimacy through frequent publication of OSS. In some cases, other units in the organizations followed suit. To nurture these advantageous cultural beliefs, public administrators should provide opportunities for individuals in development units to enhance their professional skills and grant more autonomy to these units to encourage continued exploration of innovative ways to implement technology. Developing change and risk management plans would help units adapt incrementally to change when responding to new practices in routine work. Finally, public administrators should consider how the current scope of work aligns with the FSCP to facilitate more effective ways to incorporate the policy.

Public administrators should encourage more and more varied public engagement. Public engagement is essential for information sharing because it dismantles organizational silos and enables collective problem solving. Public administrators should practice communicating and collaborating with those outside their units – with others in the agency, outside the agency, and with the public. This type of public engagement can bring diverse individuals, especially from technical and scientific domains, together to ask questions or solve a variety of problems, which can make programs more effective (Feldman & Khademian, 2007). While there is no measure for public engagement effectiveness, this study demonstrated that more and more varied engagement through bi-directional communication, holding and attending events, and using electronic tools contributed to technology policy implementation for OSS publication.

Therefore, public administrators should create infrastructure to enable those types of public engagement activities.

Public administrators should seek to train and hire individuals with more diverse technical skills. With recognition of the Hollow State literature and the findings from this study regarding diverse skills, public administrators in government agencies need to exercise more explicit monitoring and control of contractors. Additionally, they should train current federal employees to bring technical expertise back into organizations to implement policy and to increase the publication of OSS. Federal employees should continue to develop code for the government to be released as OSS, and they should update contracts to capture code developed by third parties.

Organization Theory. Organizations are comprised of individuals in pursuit of specific goals. Using theory, researchers can explain, understand, and appreciate relationships surrounding a corresponding organizational phenomenon (Hatch, 2018). The findings of this study have implications for organizations and software development units when considering structural dimensions and organizational location.

Organizations should consider how they structure their software development units. It is imperative for organizations to structure appropriately to be responsive to policy and technology changes. This study showed that less centralization, more formalization, more differentiation, and more coordination enabled units to respond and adapt effectively to environmental changes caused by the FSCP. This study aligned with the technology-structure literature with one nuance regarding formalization. Technology-structure literature stated that organizations would be less formalized when there was an environmental disturbance, but this study found that is not the case for this sample and context. The units in this study with higher OSS publication frequency had

more formalization, because they relied on the FSCP and internal agency policy to publish software. This finding contributes to the technology-structure literature as it highlights the impact of formalization on software development teams. Furthermore, there is a difference in how technology is considered in the technology-structure literature and in this study. The technology-structure literature considers technology to be a task or the development of a product; this research viewed technology as what is done with the product after production. Further research might explore the difference between product development and product processing.

Organizations should consider where software development units reside within the larger organization. Organization location aids units in responding to task complexity and uncertainty, which enhances structural flexibility (Scott & Davis, 2007). Most units in this study were located outside the CIO office. The CIO is required to implement the FSCP but may not have a holistic view of what is happening in the organization; therefore, s/he cannot effectively influence policy implementation across the organization. Units that implement the policy and that more frequently published OSS generally resided in the metaphorical middle of the organization where they could collaborate with the CIO to understand the policy while simultaneously providing services to external customers. Autonomy develops when there are fewer hierarchical layers and an executive in the organization that leads units but permits considerable autonomy. This autonomy makes it easier to work across units and organizational boundaries as well as facilitating the increased publication of OSS (Beyer, 2009; Levy, 2010; Feld, 2016). CIOs need an understanding of their organization's OSS publishing habits, which start with accounting for which units are actually publishing code. Then CIOs can prioritize and articulate strategies towards policy implementation.

Limitations and Recommendations for Future Research

As Marshall and Rossman (2009, p. 76) noted "a discussion of these [study] limitations…reminds the reader of what the study is and is not – its boundaries – and how its results can and cannot contribute to understanding." As with all research, there are a few limitations to this study. *This study focused on a particular policy for implementing a specific technology.* Government organizations implementing technology often consider cultural and structural variables, and the findings of this research should help. OSS, however, has its own set of constructs including beliefs and organizational structural characteristics other technologies do not (e.g., hacker's ethos, communal coding). *This research utilized participants' perceptions of organizational unit activities.* "One chooses a qualitative approach to understand phenomena from the participants' perspectives and to explore and discover, in depth and context" (Marshall and Rossman (2009, p. 77). At issue here is the individual's ability to recall beliefs and actions accurately as these occurred in the past. Furthermore, these perceptions might not have adequately reflected the views of others in the same unit, other software development units, or federal organizational activities more generally. *Participants self-reported the frequency of OSS publishing in their units.* These reported publication frequencies at times contradicted what was found in the public data. The units sampled were not necessarily the units that were measured in the public data, and the participants may well have not been those who reported the public data. Finally, I suggested *possible influences or associations with publishing OSS*, which was appropriate for a first study of a new phenomenon; however, a longitudinal study might provide more evidence for possible casual relationships and additional understanding of the interaction between government policy implementation and OSS publication.

In future studies, researchers should explore aspects of each variable. The variables are part of large bodies of research. An additional research question when examining culture is: Does strength of culture improve unit effectiveness with technology policy implementation for OSS publication? Schein (1986) offered that cultural strength may or may not be correlated with effectiveness but other variables relating to culture may be important to understand as units respond to environmental impacts. Finding which cultural sub-variables are relevant for warding off or adapting to environmental impacts would also be beneficial for units looking to implement technology policy.

Further examination of public engagement could include the following question: Do effectiveness, frequency, and variability of engagement mechanisms result in ease of policy implementation? This is similar to Rowe and Gammack's (2004) call for further research into effectiveness of electronic mechanisms given certain circumstances. They argue that electronic mechanisms may prove to be valuable to units so long as they know when and how often to use them.

Further examination of the relationship among cultural beliefs, public engagement, decision-making, and diverse skills could include the following: What is the relationship among the four variables and what is the relationship between each of the variables? What are the strengths of those relationships? This could provide deeper analysis into the factors influencing publication frequency.

A consideration for further study of organizational structural dimensions, specifically formalization, is the following question: Does the sort of formalization, including guides and policies, appearing in government technology units provide license to implement new technology policy? This study contradicted previous research in the technology-structure field; it would be

beneficial to better understand why this occurred. One explanation, offered by Hempel, et al. (2012), stated that formalization is helpful when it enables units to master their work and detrimental when it forces units into compliance. Units that did not recognize the environmental pressure and focused on other priorities or that did not consider the technology to be complex regardless of environmental disturbance, did not publish OSS. Furthermore, the opposite finding with formalization occurred with units that recognized the FSCP mandate and created an internal policy to enable OSS publication.

Along with exploring the variables independently, future research could also test the theory of policy implementation for OSS publication in other contexts (e.g., technology policy implementation within educational settings). Applying the theory in various contexts would validate its veracity and could provide guidance for other non-government organizations regarding policy implementation.

Conclusion

This study examined CFO Act agencies' implementation of the Federal Source Code Policy as indicated by the extent of OSS publication. The theoretical framework – including cultural beliefs, public engagement, structural dimensions, and organizational location – provided the foundation on which this inquiry began. Through the use of grounded theory, I expanded the theoretical framework to include sub-categories generated through iterations of comparative analysis and theoretical sampling. Exploring why some units published OSS and others did not yields a tentative descriptive theory of policy implementation for OSS publication, ultimately permitting publication of the People's Code.

Appendices

Appendix A: Chief Financial Officer (CFO) Act Agencies

As listed on the CFO Council website (About the Chief Financial Officers Council, 2017).

Department of Agriculture
Department of Commerce
Department of Defense
Department of Education
Department of Energy
Department of Health and Human Services
Department of Homeland Security
Department of Housing and Urban Development
Department of the Interior
Department of Justice
Department of Labor
Department of State
Department of Transportation
Department of the Treasury
Department of Veterans Affairs
Environmental Protection Agency
National Aeronautics and Space Administration
Agency for International Development
General Services Administration
National Science Foundation
Nuclear Regulatory Commission
Office of Personnel Management
Small Business Administration
Social Security Administration

Appendix B: Github.Com Metadata of Open Source Software (OSS) Published by CFO Act Agency

Agency	Total Repos	Repos pre FSCP	Repos post FSCP
Nuclear Regulatory Commission	0	0	0
Department of Housing and Urban Development	2	0	2
Department of Education	4	4	0
Agency for International Development	8	8	0
Social Security Administration	8	0	8
Department of Homeland Security	10	4	6
Office of Personnel Management	13	6	7
National Science Foundation	16	0	16
Small Business Administration	21	1	20
Department of the Treasury	22	11	11
Department of Justice	32	14	18
Department of Transportation	37	13	24
Department of Agriculture	40	38	2
Department of Labor	72	53	19
Environmental Protection Agency	144	49	95
Department of State	265	189	76
Department of Energy	307	155	152
Department of Commerce	363	240	123
Department of Veterans Affairs	370	203	167
National Aeronautics and Space Administration	411	186	225
Department of Health and Human Services	420	183	237
Department of Defense	449	270	179
Department of the Interior	993	555	438
General Services Administration	1788	1065	723

Repos = code repositories.

Metadata collected in November 2018 by modifying Irizarry's (2018) Python scripts with a Jupyter Notebook. Agencies grouped in thirds by total code repositories for easier identification and separation of agencies who minimally, intermediately, and frequently publish OSS.

Appendix C: Interview Guide

Thank you for agreeing to participate in this elite interview. The interview will be semi-structured where questions vary based on information you share and last approximately one hour. While I have your consent to participate in this study (verbally or in writing), I want to confirm that you have permission, based on your role within your organization or from your supervisor, to discuss your organizations' software development practices. Again, all data will be collected confidentially and no identifying data will be released without your written permission.

Background

- What is your current role (e.g., software developer, manager, architect)? Describe the work you do in your unit?
- What is the primary function of your unit (e.g., software development, strategy, consulting, HR)? Describe the work of your unit.

Cultural Beliefs - How do unit beliefs affect OSS publication?

- What are your unit's views on consuming open source software?
- Based on public data (e.g., GitHub.com, Code.gov) it looks like your agency publishes code [minimally, moderately, frequently].
    - Does your unit publish code publicly?
        - If yes, how much [minimally, moderately, frequently] and does your unit think this is an appropriate amount?
        - If no or a little, what are the reasons why your unit does not or minimally publish code publicly?
    - Does your unit use a public coding platform with open repositories?
- Describe your unit's goals with regards to software development (e.g., share, collaborate, be open, open by default, set an example, deliver product, adhere to standards).

Public Engagement - How do unit's practicing public engagement impact OSS publication?

- Describe how your unit uses collaboration tools externally (e.g., touchpoints, tools, frequency).
- Describe why your unit uses collaboration tools externally (e.g., knowledge sharing, policy creation, acquisition creation, code development).
- Does your unit host events, participate in events, or seek contributions from others outside the unit when developing software?
    - Describe how your unit seeks input into software development.
    - Describe why your unit seeks input into software development.

Structural Dimensions - How do structural dimensions, specifically technology-structure, influence OSS publication?

- Formalization: Describe how priorities and tasks are created and assigned in your unit. [Are they established as a result of roles, rules, and procedures or through group participation?]
- Centralization: Describe decision-making in your unit pertaining to software development.

- Differentiation: Describe how your unit is organized. [Division of labor, variation of skills]
- Coordination: Describe your unit's communication and collaboration efforts when developing software. [Formal and informal communication, frequency.]

Organizational Location - How does organizational location, or where the unit is located based on organizational form, affect OSS publication?

- Describe where your unit is located in the agency. [Primary unit (e.g., acquisition, HR) or support an external function]
- Describe the autonomous nature of your unit.

While I have your consent to participate in this study (verbally or in writing), I want to confirm that you have permission, based on your role within your organization or from your supervisor, to provide documents pertaining to your organizations' software development practices. Again, all data will be collected confidentially and no identifying data will be released without your written permission.

Artifact Collection

- Can you provide me documents regarding your unit's software development practices (e.g., software development processes, policies)?
- Can you provide me documents pertaining to your organization structure and operating procedures (e.g., organizational chart, policies for work)?

Miscellaneous

- Do you have any questions of me regarding procedures, process, or research?
- Can you refer me to a colleague at your agency or another agency that you think would be helpful for my research?

References

10 Open Source Projects to Try Over the Holidays. (2017). Retrieved Jan. 16, 2020, from https://medium.com/codedotgov/10-open-source-projects-to-try-over-the-holidays-3118a992081a.

18F: Open source policy. (2018). Retrieved Dec. 12, 2018, from https://18f.gsa.gov/open-source-policy/.

Aberbach, J. & Rockman, B. (2008). Conducting and coding elite interviews. Political Science and Politics, 35(4), 673-676.

Applegate, L. et. al. (1999). Business fundamentals as taught at the Harvard Business School: Information technology for managers. Harvard Business School Publishing.

All about Open Source. (2012). Retrieved April 2, 2018, from https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/78959/All_About_Open_Source_v2_0.pdf.

Andreessen, M. (2011). Why sofware is eating the world. pp. 1–7. Retrieved Jan. 13, 2020, from https://www.wsj.com/articles/SB10001424053111903480904576512250915629460.

Bacharach, S. B., & Aiken, M. (1977). Communication in administrative bureaucracies. The Academy of Management Journal, 20(3), 365–377.

Balter, B. (2018). Why isn't all government software open source? Retrieved Jan. 13, 2020, from ben.balter.com website: https://ben.balter.com/2014/08/03/why-isnt-all-government-software-open-source/.

Barrett, D. (2007). Cultural changes key to reducing barriers to open source software. Retrieved from SIGNAL blog, http://www.afcea.org/content/?q=cultural-changes-key-reducing-barriers-open-source-software.

Beyer, K. (2009). Grace Hopper and the invention of the information age. MIT Press.

Boisnier, A., & Chatman, J. A. (2002). The Role of Subcultures in Agile Organizations. Retrieved Jan. 13, 2020, from https://www.researchgate.net/publication/265206215_The_Role_of_Subcultures_in_Agile_Organizations.

Bozeman, B., & Bretschneider, S. (1986). Public management information systems: Theory and prescription. Public Administration Review, 46(Special Issue), 475-487.

Boyd, A. (2019). Why It's So Hard to Write Federal Technology Policy. Retrieved Jan. 13, 2020, from https://www.nextgov.com/policy/2019/09/why-its-so-hard-write-federal-technology-policy/160100/.

Brudney, J., O'Toole, L., & Rainey, H. (2000). Introduction: Public management in an era of

complexity and challenge. In J. Brudney, L. O'Toole, & H. Rainey (Eds.), Advancing Public Management: New Developments in Theory, Methods, and Practice (p. 320). Georgetown University Press.

Burton, R. A. (2004). M-04-16, Software Acquisition.

Cameron, Kim S., Ettington, D. R. (1988). The Conceptual foundations of organizational culture. In. Smart, J. C. (Ed.), Higher Education: Handbook of Theory and Research, Vol. 4, pp. 356–396.

Caudle, S. L. (1987). Federal Information Resources Management: Bridging Vision and Action. Washington, DC: National Academy of Public Administration.

Ceruzzi, P. E. (2012). Computing: A concise history. Cambridge: The MIT Press.

Charles, F., & Ferguson, C. (2005). How linux could overthrow microsoft. MIT Technology Review, 108(6), 64.

CIO Management Responsibilities Remain a Challenge for Most Agencies. (2018). Retrieved Jan. 13, 2020, from https://blog.gao.gov/2018/11/27/cio-management-responsibilities-remain-a-challenge-for-most-agencies/.

Constantine, L. (2007). The open-source solution. MIT Technology Review, (January/February), 26.

Cooke, R., & Szumel, J. (1993). Measuring normative beliefs and shared behavioral expectations in organizations: The reliability and validity of the organizational culture inventory. Pyschological Reports, 72, 1299–1330.

Corbin, J., & Strauss, A. (1990). Grounded theory research: Procedures, canons, and evaluative criteria. Qualitative Sociology, 13(1), 3–21.

Creswell, J. W. (2009). Research design : qualitative, quantitative, and mixed methods approaches. Sage Publication.

Creswell, J. W., & Miller, D. L. (2000). Determining validity in qualitative inquiry. Theory Into Practice, 39(3), 124–130.

Dearden, J. (1972). MIS is a mirage. Harvard Business Review, 50(1), 90-99.

Dixit, A. (2012). Bureaucracy, its reform and development. Review of Market Integration, 4(2), 135–157.

Feld, E. (2016). The Government GitHub Ecosystem. Retrieved November 7, 2017, from https://emanuelfeld.github.io/blog/2016/04/27/government-github-ecosystem.html.

Feldman, M. S., & Khademian, A. M. (2007). The role of the public manager in inclusion: Creating communities of participation. Governance, 20(2), 305–324.

Fry, L. W. (1982). Technology-structure research: Three critical issues. Academy of Management Journal, 25(3), Vol 25, No 3, pp. 532–552.

Gates, B. (1976). Bill Gates – An Open Letter to Hobbyists. Retrieved February 14, 2018, from https://genius.com/Bill-gates-an-open-letter-to-hobbyists-annotated.

Germain, J. M. (2014). The U.S. Government's Tenuous Relationship With Open Source. Retrieved Dec. 31, 2019, from Linux Insider website: https://www.linuxinsider.com/story/81348.html.

Gibson, B., & Hartman, J. (2014). Rediscovering grounded theory. London: Sage Publication.

GitHub is How People Build Software. (2018). Retrieved June 3, 2018, from https://github.com/about.

Glaser, B. (1978). Theoretical sensitivity: Advances in the methodology of grounded theory. Sociology Press.

Glaser, B., & Strauss, A. (1967). The discovery of grounded theory: Strategies for qualitative research (Reprinted). New Jersey: Aldine Transaction.

Government-wide Information Technology Agency Summary. (2017). Retrieved Dec. 31, 2019, from ITDashboard.gov website: https://www.itdashboard.gov/drupal/summary/000.

GraphQL API v4: Overview. (2018). Retrieved Jan. 13, 2020, from https://developer.github.com/v4/.

Gregor, S. (2006). The nature of theory in information systems. MIS Quarterly: Management Information Systems, 30(3), 611–642.

Hage, J., Aiken, M., & Marrett, C. B. (1971). Organization structure and communications. American Sociological Review, 36(5), 860–871.

Hall, A. (1997). Tacit Culture and Change: A Model of Change Constructed from Institutional Assumptions and Beliefs. Retrieved Dec. 31, 2019, from https://vtechworks.lib.vt.edu/handle/10919/30565.

Harris, M., & Raviv, A. (2014). Organization design. Management Science, 48(7), 852–865.

Hatch, M. (2018). Organization theory: Modern, symbolic, and postmodern perspectives (4th ed.). Oxford: Oxford University Press.

Hatch, M. J. (1997). Organization theory: Modern, symbolic, and postmodern perspectives (1st ed.). Oxford: Oxford University Press.

Hempel, P., Zhang, Z., & Han, Y. (2012). Team empowerment and the organizational context: Decentralization and the contrasting effects of formalization. Journal of Management, 38(2), 475-501.

Heydebrand, W. (1989). New organizational forms. Work and Occupations, 16(3).

Holden, S. (2003). The Evolution of Information Technology Management at the Federal Level: Implications for Public Administration. In G. D. Garson (Ed), Public information technology: Policy and management issues (pp. 53-73). Hershey, PA: Idea Group Publishing.

Holton, J. & Walsh, I. (2016). Classic grounded theory: Applications with qualitative and quantitative data (1st ed.). London: Sage Publications.

How Agencies Build Software. (2018). Retrieved Jan. 13, 2020, from https://government.github.com/.

Irizarry, F. (2018). Code.gov Stats with Jupyter Notebook. Retrieved Jan. 13, 2020, from https://github.com/GSA/code-gov-stats-jupyter-notebook.

Isaacson, W. (2014). The innovators: How a group of hackers, geniuses, and geeks created the digital revolution. New York: Simon & Schuster.

Johnson, N. B., & Sears, L. (2017). 7 Open Sourced Myths Debunked. Retrieved from https://go.govloop.com/open-source-myths.

Jones, J. (2018). Happy second birthday to Code.gov. Retrieved Nov. 15, 2018, from https://github.blog/2018-11-09-happy-birthday-codegov/.

Kadir, N. (2015). Bureaucratizing Participation: Stakeholders' Perceptions of the Administrative Rules Governing Public Participation in the Hampton Roads Transportation Planning Organization. Retrieved from https://vtechworks.lib.vt.edu/handle/10919/76671.

Khademian, A. M. (2002). Working with culture: How the job gets done in public programs. CQ Press.

Kim, S. & Thompson, S. (2013). Lessons for software development ecosystems: South Korea's e-government open source initiative. MIS Quarterly Executive, 12, 93-108.

Lawrence, P., & Lorsch, J. W. (1967). Differentiation and integration in complex organizations. Administrative Science Quarterly, 12, 1–47.

Lee, I. (2018). The case for open source software. Retrieved Nov. 16, 2018, from https://18f.gsa.gov/2018/07/12/the-case-for-open-source-software/.

Levy, S. (2010). Hackers: Heroes of the computer revolution. O'Reilly Media.

LLNL Software Portal. (2020). Retrieved from https://software.llnl.gov/.

Locke, K. (1996). Rewriting the discovery of grounded theory after 25 years? Journal of Management Inquiry, 5(3), 239–245.

Mackenzie, K. D. (1978). Organizational structures. AHM Pub. Corp.

Mann, C. C. (2002). Why software is so bad. MIT Tech, 105(6).

Mansfield, R. (1973). Bureaucracy and centralization : An examination of organizational structure. Administrative Science Quarterly, 18(4), 477–488.

Marshall, C., & Rossman, G. B. (2011). Designing qualitative research. Sage Publication.

Martin, J. (1992). Cultures in organizations. New York: Oxford University Press.

Martin, J., & Siehl, C. (1983). Organizational culture and counter culture: An uneasy symbiosis. Organizational Dynamics, 12(3), 52–64.

Milward, H. B., & Provan, K. G. (2000). Governing the hollow state. Journal of Public Administration Research and Theory, 10(2), 359-379.

Mintzberg, H. (1983). Structure in fives: Designing effective organizations. Peason Publishing.

Mockus, A., Fielding, R. T., & Herbsleb, J. D. (2002). Two case studies of open source software development: Apache and mozilla. ACM Transactions on Software Engineering and Methodology (TOSEM), 11(3), 309–346.

National Defense Authorization Act for Fiscal Year 2018: H.R. 2018. (2018). Retrieved from https://www.congress.gov/bill/115th-congress/house-bill/2810.

Nilsson, N. (2014). Understanding beliefs. MIT Press.

Nystrom, P., & Starbuck, W. (1984). Theoretical observations on applied behavioral science. The Journal of Applied Behavioral Science, 20(3), 277–287.

Offenbacker, B. (2010). Inclusive Management in Action: An International Study of Public Engagement. Retrieved from https://vtechworks.lib.vt.edu/handle/10919/28276.

Open Source Software and the Public Sector. (2009). Retrieved Jan. 13, 2020, from https://ma.ellak.gr/documents/2014/06/open-source-software-and-the-public-sectora-danish-perspective.pdf.

Paperwork Reduction Act of 1980, 44 U.S.C. 3501 et. seq (1980).

Passingham, J. (2016). Open Source Software Finds Its Sweet Spot in the Public Sector. Retrieved Dec. 31, 2019, from Public Sector Executive website: http://www.publicsectorexecutive.com/Public-Sector-News/open-source-software-finds-its-sweet-spot-in-the-public-sector.

Pettigrew, A. M. (1979). On studying organizational cultures. Administrative Science Quarterly, 24(4), 570–581.

Roberts, J. A., Hann, I. H., & Slaughter, S. A. (2006). Understanding the motivations, participation, and performance of open source software developers: A longitudinal study of the apache project. Management Science, 52(7), 984–999.

Rosenberg, S. (2007). Anything You can do - I can do meta. MIT Technology Review, 110(1).

Rowe, G., & Frewer, L. J. (2005). A typology of public engagement mechanisms. Science Technology and Human Values, 30(2), 251–290.

Rowe, G., & Gammack, J. G. (2004). Promise and perils of electronic public engagement. Science and Public Policy, 31(1), 39–54.

Sabatier, P. (2007). Theories of the policy process (2nd ed.). Westview Press.

Sabatier, P., & Mazmanian, D. (2005). The implementation of public policy. Policy Studies Journal, 538–560.

Sapienza, A. (1985). Believing is seeing: How culture influences the decisions top managers make. In Kilmann, R., Saxton, M., & Serpa, R. (Eds.), Gaining control of corporate culture (pp. 66–83). San-Franscisco: Jossey-Bass.

Sathe, V. (1983). Implications of corporate culture: A manager's guide to action. Organizational Dynamics, 12(2), 5–23.

Schein, E. H. (1986). Coming to a new awareness of organizational culture. Sloan Management Review, 25(2), 3-16.

Schein, E. H. (2010). Organizational culture and leadership. San Francisco, CA: Jossey-Bass.

Scott, T. (2017). Why is there so much customized software in the federal government? Retrieved December 31, 2019, from https://fcw.com/articles/2017/11/27/comment-tony-scott-gov-custom-code.aspx.

Scott, T., & Rung, A. (2016). M-16-21 - Federal Source Code Policy: Achieving Efficiency, Transparency, and Innovation through Reusable and Open Source Software. Retrieved Dec. 31, 2019 from https://obamawhitehouse.archives.gov/sites/default/files/omb/memoranda/2016/m_16_21.pdf.

Scott, W. R., & Davis, G. F. (2007). Organizations and organizing : rational, natural, and open system perspectives. New York: Routledge.

Sharing America's Code. (n.d.). Retrieved Jan. 17, 2020, from https://code.gov/.

Sorge, A. (1989). An essay on technical change: Its dimensions and social and strategic context. Organization Studies, 10(1), 23-44.

Strauss, A. (1987). Qualitative analysis for social scientists. New York: Cambridge University Press.

Strauss, A., & Corbin, J. M. (1994). Grounded theory methodology. In N. K. Denzin & Y. S. Lincoln (Eds.), Handbook of qualitative research. Thousand Oaks: Sage Publications, Inc.

Strauss, A. & Corbin, J. M. (1998). Basics of qualitative research: Techniques and procedures for developing grounded theory (2nd ed.). Thousand Oaks, CA: Sage Publications, Inc.

Strenio, E., & Ashida, A. (2019). Human-centered design for IT centralization, part 5 - Centralization gone right: A case study on the U.S. Web Design System. Retrieved Jan. 16, 2020, from https://18f.gsa.gov/2019/05/21/centralization-gone-right-a-case-study-on-uswds/.

The Open Source Definition (Annotated). (n.d.). Retrieved November 7, 2017, from opensource.org website: https://opensource.org/docs/definition.php.

Thompson, J. (2017). Organizations in action. New York: Routledge.

Tunstall, W. B. (1986). The Breakup of the bell system: A case study in cultural transformation. California Management Review, 28(2), 110–124.

United States Web Design System. (2020). Retrieved from https://designsystem.digital.gov/.

WALKOFF. (n.d.). Retrieved Jan. 16, 2020, from https://nsacyber.github.io/WALKOFF/.

Water Network Tool for Resilience (WNTR) User Manual. (2019). Retrieved Jan. 16, 2020, from https://cfpub.epa.gov/si/si_public_record_report.cfm?Lab=NHSRC&dirEntryId=337793.

Weber, S. (2004). The success of open source. Cambridge: Harvard University Press.

Weber, M. (2017). Essays in sociology; Translated, edited, and with an introduction. Reprint of 1946 book published by Oxford University Press.

Weiss, R. S. (1994). Learning from strangers : The art and method of qualitative interview studies. Free Press.

Woodward, J. (1981). Industrial organization: Theory and practice (2nd ed.). Oxford: Oxford University Press.