Intellectual Property and Architecture: New Research on How to Avoid Lock-In

Maj Chris Berardi, USAF, and Bruce Cameron

Intellectual property lock-in is a wicked problem that is particularly pervasive in the Department of Defense (DoD) and other market structures characterized by one buyer and many sellers (i.e., monopsony). However, little research exists on the mechanisms or action(s) that induce intellectual property lock-in. This work postulates that the conjuncture of architecture and intellectual property is one such mechanism that can induce lock-in. To investigate links between intellectual property, architecture, and lock-in, the authors formulated and executed an intermediate-N fuzzy-set Qualitative Comparative Analysis research approach, using 14 DoD software cases representing over 34 million lines of code. Within the sample, they found an accessible intellectual property architecture—absent any other explanatory conditions—is sufficient to avoid lock-in. The authors suggest that software architectures with small core groups are more conducive to lock-in, particularly when coupled with open source or software to which the DoD has an unlimited right comprising more than 50% of the core intellectual property.

Two-line Summary: This article describes the formulation and execution of an intermediate-N fuzzy-set Qualitative Comparative Analysis research approach, using 14 DoD software cases representing over 34 million lines of code, to analyze conditions that result in intellectual property lock-in.

Keywords: Intellectual Property Lock-in, Data Rights, Technical Data, Competition, Qualitative Comparative Analysis

Intellectual property lock-in is a wicked problem. Rittel and Webber (1973), professors of design and urban planning, defined wicked problems as a class of problems that is not easy to define, has innumerable causes, is tough to describe, and has no single correct answer. A recent study suggests that lacking rights to patents or intellectual property was listed by the Department of Defense (DoD) as the motivation for single-sourced contracts valued at a combined $6.3 billion for Fiscal Years 2008–2015 (Berardi, Cameron, & Crawley, 2017). This finding suggests that a substantial number of program managers are concerned enough with intellectual property to preclude competition—an action that has well-defined implications for pricing and innovation (Stucke, 2013).

Background

In practice, intellectual property lock-in is often used to describe a situation where some manner of intellectual property is used to reduce or preclude competition; the phrase is rarely defined in literature.

William Anderson (2003, p. 38), in an article for the *Public Contract Law Journal*, recounts the contracting actions of Captain Charles Wallace, who entered into a procurement contract with Wilbur and Orville Wright for delivery of the first "heavier-than-air flying machine" in 1908. Within this seminal contract was terse, yet effective, intellectual property language that required no patent rights be transferred to the government as a result of the sale and required delivery of drawings under the promise that they would not be shown to competitors. In total, the entire contract was printed on two legal-sized pages.

This contracting and intellectual property landscape has changed dramatically since the Wright Brothers' contract. A review of articles in public contract law journals reveals a chorus of both practitioners (Anderson, 2003; Dix, Lavallee, & Welch, 2003; Dungan, 2010; Sharp, 2003; Sidebottom, 2003; Simchak, 2003) and academics (Larson, 2007; Lavenue, 1998; Lill, 2009) who assert that the state of intellectual property regulation is either too difficult to understand or too protective of government interests. Furthermore, most argue that the confusing labyrinth of statute and policy deters innovation from large and small companies; companies would rather "forgo contracting with the Federal Government rather than risk the misuse or loss of valuable intellectual property" (Trend, 2005, p. 290).

Over the years, the U.S. Government and defense industry have asserted a number of potential causes of lock-in, from insufficient competition in a market to a monopoly on key intellectual property rights. However, recent amendments to 10 U.S.C. § 2320 are significant because they mark the emergence of theoretical principles of architecture in DoD intellectual property statute. For example, the National Defense Authorization Act of 2012[1] (NDAA 2012) authorized the DoD to release proprietary data that are commonly referred to as "interface data," which were defined in the Act of 2012 as data "necessary for the segregation of an item or process from, or the reintegration of that item or process (or a physically or functionally equivalent item or process) with, other items or processes." Additionally, the National Defense Authorization Act of 2015[2] (NDAA 2015) required the DoD to develop standards and define architectures necessary to enable open systems approaches. NDAA 2015 further mandated DoD acquisition programs and contract structures employ Open Systems Architectures to the maximum extent practicable, which in itself was suggestive that "closed architectures" were responsible for lock-in.

The concept of system architecture originates primarily in the seminal work of Herbert Simon (1962), wherein he proposed the theories of hierarchy and "near decomposability." At the most basic form, architecture formalizes abstract patterns of interfaces between components. However, Ulrich (1995, p. 2) offers a more precise three-part definition of architecture: "1) the arrangement of functional elements, 2) the mapping from functional elements to physical components, and 3) the specification of interfaces among physical components." For any given product, there could be many different arrangements of functional elements, and subsequently even more mappings from functional elements to physical components, which satisfy the functional requirements (MacCormack, Rusnak, & Baldwin, 2006). Specifically, a product may be designed for ease of manufacturing or efficient development as opposed to maximized performance. Consistent with the seminal teachings of architecture, Baldwin and Henkel (2014) argue an architecture may also be organized to protect intellectual property—a concept referred to as *Intellectual Property Modularity*.

One conceptual model that has been used to explain lock-in is the "bottleneck" model (Jacobides, Knudsen, & Augier, 2006; Jacobides & Tae, 2015; Jacobides, Velosa, & Wolter, 2013; Langlois, 2002; Pisano & Teece, 2007). In this context, a bottleneck begins as a nontrivial technical problem. For example, in a traffic system, if there is no bridge over a river, goods would need to be off loaded from trucks onto ferries and reloaded onto rail cars or different trucks on the other side of the river. Once an organization identifies a bottleneck and creates an effective solution (i.e., value creation), it may then combine that solution with property rights to extract a stream of rents (i.e., value capture). Although this example represents a bottleneck as a technical problem, Baldwin (2015, p. 11) extends this definition to strategic bottlenecks, which require "(1) a unique solution to an underlying technical bottleneck; plus (2) control over access to the solution." Using the analogy of the bridge over a river (technical bottleneck), an organization must first build a bridge to create a strategic bottleneck, then exclude others from using it unless they pay a toll. It is important to point out that the bridge in this example must also be nonsubstitutable (i.e., it cannot be built next to another bridge).

Given a variety of statute-espousing solutions to lock-in, and existing conceptual models that postulate both architecture as a source of lock-in and intellectual property as a source of lock-in, this article seeks to examine the mechanisms by which intellectual property leads to lock-in. We focus specifically on the acquisition of software, representing a growing proportion of system cost and a challenging intellectual property environment. First, we review the research and legal literature on intellectual property and lock-in in the context of military acquisition. We then provide an overview of the research methods that were employed, followed by a discussion of the results testing for sufficient conditions for lock-in. We conclude with the implications for practice.

Literature

The literature review herein departs slightly from a traditional research literature review in that legal literature is also included. Legal literature differs from research literature in that it does not employ a research-based methodology. Quantitative results are few and none of the arguments are backed by statistical significance. Nonetheless, the inclusion of legal literature is paramount because that is the primary domain of DoD intellectual property regulation. The authors highlighted in the following discussion are the foremost thinkers on the topic and present arguments advancing the practice of intellectual property regulation using case law and statute as the foundation.

Legal Literature

All legal literature reviewed, with the exception of Lavenue (1998), unanimously argues that intellectual property regulations are overly complex. The arguments against overly complex regulations will be addressed shortly, but prior to beginning, it is important to investigate Lavenue's (1998) counterargument.

> With an infancy in the patent clauses of the 1940s, and following various incantations over the last fifty years, the 1995 data rights regulations represent not only the most considered and precise regulations for the protection of computer software and technical data, but also the most fair. (p. 68)

Lavenue argues that the regulations are not overly complex, but rather "precise." He argues this precision is necessary for the type of work contracted under the Defense Federal Acquisition Regulation Supplement (DFARS) and also drives equity amongst interested parties. Most literature that criticizes the complex nature of DoD intellectual property regulations relates to commercial intellectual property statutes. Lavenue makes the point that the needs of the military are drastically different from the commercial industry and that the regulations are a manifestation of this difference. Lavenue supports this claim by citing an example where the military must frequently make repairs in austere locations far away from the contractor who built the system; in such cases, technical data are often a necessity to accomplish the military objective. The civilian intellectual property schema offers no consideration of such exigent circumstances, which is why it may be less complex.

Lavenue's arguments are rational, but pose the question: What is the correct balance between the specificity needed to handle situations unique to the military, but not overly burdensome to those contractors who wish to do business with the DoD? Identifying this optimal balance is the topic of much of the remaining legal literature.

Lazure and Church (1954), Sidebottom (2003), Trend (2005), Mazour (2009), and Murray (2012) argue that the government is, at times, limited by statute. The premise of this argument revolves around the language in the FY07 NDAA, which created a statutory requirement for program managers to foresee long-term data rights needs. Although well intentioned, compliance is difficult because no methodology exists to forecast data rights needs. To comply with this statute, program managers may attempt to acquire many more data rights than are truly needed. All six authors listed above cite this as the prime reason why tension exists between industry and government, arguing that if this statutory requirement were relaxed, it would allow the DoD to acquire a smaller set of rights at later dates in the contract. Theoretically, this argument is valid; deferring the acquisition of data rights to the latest point possible will allow for purchase of the minimum possible rights. But in so doing, it also forfeits most of the DoD's negotiating power.

Interestingly, all of the authors addressed this criticism under the lens of timing the purchase of data rights to acquire the least amount of rights necessary, but none examined it from the perspective of improving the methodology to determine which rights are needed.

Anderson (2003), Sidebottom (2003), Trend (2005), and Mazour (2009) argue that the regulations need to either be simplified or unified across all government agencies. This criticism is rooted in the hierarchical nature of the Federal Acquisition Regulation (FAR). Depending on the government agency with which a contractor is doing business, the intellectual property regulations may vary. For example, a contractor selling identical satellite technology to the Air Force and NASA is subject to the regulations in the FAR, DFARS, and Air Force Federal Acquisition Regulation Supplement for one contract, and subject to the FAR and NASA FAR Supplement in another. Although the product and intended use may be identical, the acquisition regulations are different. These differences make doing business with the government burdensome because of the different rules for each potential market. The authors make the point that if the system were unified, it would reduce complexity as well as tension between industry and government. The argument is rational, but it does not consider Lavenue's (1998) counterargument that precision in regulation is necessary for specific agency objectives. It is possible that no agency objectives are compelling the multiple sets of published regulations, but without subsequent analyses, determining the validity of this argument is difficult.

Lill (2009), Mazour (2009), and Dungan (2010) argue that the DoD's current system of marking intellectual property is overly onerous, and simplifying the requirement would attract more potential innovators. Under current regulations, a contractor can protect data in transactions with the government two ways: mark the data using restrictive legends or withhold the data entirely (Lill, 2009). A restrictive legend is designed to notify the government of its rights to any information contained in a deliverable. In regulation, a contractor can fail to comply in any one of three ways: (a) by not marking data, (b) by using unauthorized marking, or (c) by failing to correctly mark data. If a contractor fails to mark data, the government assumes the data are furnished with an unlimited right. If a contractor marks the data, but is unauthorized to do so under the DFARS or FAR, the government may remove or ignore the markings. Lastly, if a contractor marks the data incorrectly or uses a nonconforming marking, the government may ignore the marking. In practice, the execution of such rules is typically not as definitive, and contractors are often given an opportunity to correct markings even in cases where the regulations do not require such consideration. However, the nature of these regulations tends to stir a sense of wariness among companies doing business with the DoD because of the potential that they will make an unintentional “gift” of intellectual property simply by failing to mark data correctly. The authors argue this marking requirement is a significant disincentive to promoting innovation and a large issue driving tension.

Trend (2005), Mazour (2009), and Dungan (2010) argue that the DoD should eliminate the unlimited right. The authors argue that the differences between the government purpose right and unlimited right are subtle and that the government purpose right is adequate for all purposes while still protecting a contractor's data. Mazour (2009, p. 686) goes even further in arguing against the unlimited right. “The only justification for the government to have such broad rights is to provide it with an excuse if it mishandles the technical data.”

These arguments are valid, but similar to other arguments, fail to address Lazour's military necessity counterargument. At the outset of a contract, the DoD is not capable of foreseeing all possible use cases for a system or military objectives. In such cases where the unlimited right is purchased at the outset of a contract, the right provides an avenue to hedge against uncertainty. While it is possible that a government purpose right may suffice in such situations, none of the authors address the counterargument.

Sharp (2003) and Murray (2012) take a different approach by arguing that regulations are not necessarily overly complex, but that education in the regulations is lacking. This is a salient point. Entire legal institutions and firms abound whose sole purpose is to either instruct or assist in arbitrating commercial intellectual property matters, but the same institutional instruction and arbitration capacity does not exist for DoD intellectual property matters. In light of this dearth of knowledgeable legal entities, fewer experts are available for consultation on DoD intellectual property matters. Also, and outside of legal expertise, few educational opportunities exist for practitioners.[3] This point is also observed in the literature reviewed. Over half of the scholarly legal articles reviewed were andrological in nature (Anderson, 2003; Baker, 2011; Dix et al., 2003; Hinrichs, 1967; Larson, 2007; Lavenue, 1998; Lazure & Church, 1954; Pages, 2013; Pickarz, 2015; Sharp, 2003). Their sole purpose was to help advance the understanding of DoD intellectual property policy versus advance policy with an argument. Some made the occasional closing argument for simplifying regulation at the conclusion, but these arguments were not the objective of this article. This is further evidenced by many of the article titles, which are similar to Sharp's (2003) title, *A Layman's Guide to Intellectual Property in Defense Contracts* (2003). An additional set of literature also serves as a “how-to" guide for contractors to protect their data (Baker, Haque, Mccarthy, & Raddock, 2015; Duberstein, 1988; Madden, Marvin, & Reams, 2006; Simchak, 2003; Simchak & Vogel, 1994). These are generally characterized by the title, *Government Data Rights Challenges: Tips on How to Avoid & Respond* (Baker et al., 2015). DoD intellectual property experts in both academia and practice are few. The complex nature of regulations as well as the lack of expertise is part of the problem. However, arguing causality between these two variables is impossible, as fixing one would likely not fix the other.

Larson (2007) and Mazour (2009) argue for the creation of a research and development (R&D) funding *watermark*. This would create a threshold in mixed funding situations where the government would have to fund a certain percentage before a government purpose right would vest. The premise of this argument is that a private contractor could fund 99% of the development costs and still have to relinquish a government purpose right. Although seemingly valid, it is important to note that the inverse of this situation is also possible, where the government could fund 99% and still only receive a government purpose right instead of an unlimited right. The difficulty with establishing a funding threshold is that it incentivizes contractors and the government to game the system—in effect, staying below or above the watermark. Larson (2007, p. 199) acknowledges public criticism of watermark efforts because, without some form of license, the public could pay twice for inventions—“once when taxes are used to fund R&D and again via high prices due to a monopoly on the invention.” Any solution would have to balance these concerns.

Research Literature

The previous summary of legal literature is now juxtaposed against a much smaller body of research-based literature. Research-based literature represents those contributions that are based on a formal, research-based methodology.

Muzzelo and Arndt (2014) hypothesized that government ownership of data rights makes no difference on the transition of technology projects from lab to field. To test this hypothesis, they drew data from the 71 Army Science & Technology (S&T) Advanced Technology Development programs. Of the initial 71 programs, 57 were selected for final analysis. These 57 programs were then scored as to whether a project successfully transitioned from the S&T community to a Program of Record (PoR) according to the category of government rights (i.e., limited, government purpose, or unlimited). Those that transitioned were given a value of 1 and those that did not transition were given a value of 0. Using an ANOVA statistic test, Muzzelo and Arndt were able to reject the null hypothesis ($p = 0.024$) and accept the alternate hypothesis—that government ownership of technical data rights makes a difference in technology transition.

Although an interesting finding, the generalizability of Muzzelo and Arndt's conclusions is questionable. First, the sample collected only represents programs from the Army. This is problematic because intellectual property regulations are executed with minor differences between the Services. No action was taken to control for these differences. Furthermore, Muzzelo and Arndt (2014, p. 640) do not explain the methodology used to reduce the sample from 71 to 57. They simply state, "excluding the data determined to be meaningless." Without understanding this process, the impact, if any, that this sampling had on the end result is difficult to determine. Lastly, assuming the research was entirely valid, the results only speak to the transition of technology from the S&T community to PoR. Consequently, these results are likely not generalizable to technologies that do not start in the S&T community, which is the majority of DoD R&D efforts.

Liedke and Simonis (2014) analyzed contract data from the U.S. Army Tank-automotive and Armaments Command (TACOM) FY12 Annual Competition Report to determine if the government's lack of data rights impacted the Army's ability to competitively acquire noncommercial hardware. In FY12, TACOM executed \$3.7 billion on a competitive basis (38% of total budget) and \$9.7 billion on a noncompetitive basis (62% of total budget). Seven out of the top 10 dollar value noncompeted contract actions cited lack of technical data rights as a primary factor in preventing competition. Liedke and Simonis' (2014) research effort is the only one that reviewed the Justification and Authorizations[4] (J&A) for each noncompetitive contract. Their research led to interesting findings, albeit with a small sample:

> Most of the J&As reviewed discussed how the government requested that the contractor provide a cost estimate for selling unlimited rights to the government for their data. In all these cases, the estimate the government received was either so high that procuring the data rights was far from being a cost-effective approach, or the contractor refused to even provide an estimate whatsoever. (p. 46)

The small, domain-specific sample collected may reduce the generalizability of Liedke and Simonis' results, but the research efforts are salient because they represent the first attempt at quantifying the impact of lacking data rights on DoD acquisition. Although it is not possible to extrapolate trends for the entire DoD based on this research, it introduces a methodology for understanding and analyzing trends in DoD noncompetitive acquisition due to lack of data rights. This work was later built upon by Berardi et al. (2017) in an attempt to offer a DoD-wide extrapolation.

Literature Gaps

The literature presented in this review was organized into legal literature and research literature, with the preponderance falling into the legal literature category. This general organization demonstrates the largest gap in DoD intellectual property literature. Addressing the gaps between categories first, the legal category and the research category are essentially silos of academic thought with little to no connection between the two fields. The result is rather limited discussions that only consider arguments that fall within each domain. For example, in the entirety of the legal authorship examined for this review, not a single author argued for a nonlegal solution (i.e., a solution that does not involve changing policy or statute). The best example of this phenomena is the argument concerning the government's limitation by statute. Six authors all agreed on the salience of the argument and six authors all made recommendations within the legal silo. The premise of this argument is that the DoD does not have the methodology to determine what rights are needed in the future. However, each author ignores advancing the methodology and seeks to remedy the argument through adding additional complexities to statute and regulation. The situation is akin to the old adage, "Everything looks like a nail when all you have is a hammer." *Consequently, the gap is a lack of cross-domain approaches, which combine legal expertise, but also engineering and business practices, in an attempt to advance the methodology of data rights forecasting and making effective business cases around such decisions.* Some of this cross-domain effort is evident in the recent NDAAs, which are beginning to target system architecture as a viable methodology.

Within each category, the gap is evidenced by the disproportionate amount of legal arguments relative to the amount of research that employs a formal, academic-based research methodology. Given the extensive, and oftentimes contentious, DoD intellectual property history, one may surmise that much study has focused on the impacts of these policies on competition levels in defense spending. However, less than five research articles address even these basic questions. The absence of such analyses and quantitative analysis is concerning because it raises the question: What data were used to inform policy decisions? This literature review concludes that gaps are twofold: (a) cross-domain approaches towards identifying methodological approaches to analyzing data rights, and (b) quantitative analyses on the impacts of intellectual property policy on DoD acquisition.

Method

To effectively investigate these literature gaps, a multiple case study design across a number of DoD software programs was used to identify a mechanism of action that would allow for the forecasting of data rights needs in software. Although DoD software programs vary widely in terms of requirements, domain, and design choices, this research uses the same basic framework for each case study. This common methodology, coupled with semistructured interviews to develop context around each case, allowed for deeper cross-case comparison and ultimately led to more meaningful conclusions on the mechanisms that induce intellectual property lock-in.

In selecting research design, social scientist Charles Ragin (1987) argues for a synthetic approach that combines aspects from case-oriented and variable-oriented approaches, which led to Ragin's development of a synthetic class of inquiry broadly termed Configuration Comparative Methods or more specifically, Quantitative Comparative Analysis (QCA). The development of QCA as a method was furthered by Ragin (1994, 2005, 2006) and Rihoux (Berg-Schlosser, De Meur, Rihoux, & Ragin, 2009; Rihoux, 2003; Rihoux & Lobe, 2009; Varone, Rihoux, & Marx, 2006). QCA adopts some of the key strengths of the qualitative approach, in that each case is considered as a complex entity (i.e., a whole), not just a collection of variables. Additionally, it relies on a rigorous set of quantitative tools (Boolean mathematics) from which to draw logic-based conclusions and reduce cases into a series of explanatory conditions.

Model Specification

Within QCA are three methodological approaches: crisp-set QCA (csQCA), multivalue QCA (mvQCA), and fuzzy-set QCA (fsQCA). Each of these approaches uses the same theoretical foundations, but executes the approach of that foundation in different ways. In csQCA, each explanatory condition is dichotomous representation of set membership ("1" signifies membership, "0" signifies nonmembership). Cases are combined across these dichotomous variables and reduced using Boolean algebra to reach a parsimonious conclusion (Rihoux & De Meur, 2009). However, problems arise in csQCA due to the information loss associated with dichotomization. Conditions that may be only somewhat similar are given the same dichotomous value, which leads to cases with identical conditions, but conflicting outcomes. Multivalue QCA deals with this information loss by removing the dichotomy requirements and allowing for a multivalue category ($X\{0; 1; 2\}$ instead of $X\{0; 1\}$). Although a small adjustment, this extension of csQCA into mvQCA reduces possible conflicts and keeps the sample space small (Cronqvist & Berg-Schlosser, 2009). The final approach is fsQCA, which draws upon fuzzy-set theory, a well-developed mathematical system of addressing partial membership in sets, to analyze conditions along a continuous interval scale (between 0 and 1) (Ragin, 2009). Although this method is more challenging computationally, it addresses many of the limitations of csQCA and mvQCA. Further, fsQCA is particularly well suited for conditions that are not easily organized into a dichotomy or multichotomy, as it permits researchers to preserve natural patterns in raw data by allowing partial set membership.

The research design chosen is a Fuzzy Set QCA (fsQCA) model (Table 1). The first condition is fuzzy representation of quality technical architecture. Fuzzy-set membership values close to "1" indicate membership in the set Quality Technical Architecture. The fuzzy membership scores are extrapolated from the core size of each case's architecture. A lower core size is associated with higher quality technical architecture (Baldwin, MacCormack, & Rusnak, 2014; Sturtevant, 2013). Conversely, fuzzy membership scores close to "0" represent nonmembership in the set of Quality Technical Architecture. The second condition, intellectual property architecture, is operationalized in an almost identical manner to quality technical architecture, where values close to "1" represent membership in Accessible Intellectual Property Architecture and values close to "0" represent nonmembership. The final input condition is unlimited rights; this condition captures the dichotomous representation of whether the DoD acquired an unlimited right at the outset of the contract ($UR\{1\}$) to the software or did not acquire an unlimited right ($UR\{0\}$). It is important to note that this is a binary representation of membership in the set of cases where the DoD acquired an unlimited right to the software code. It does not indicate anything about other classes of rights (i.e., $UR\{0\} \nrightarrow$ Restricted Rights, $UR\{0\} \nrightarrow$ Government Purpose Rights). Finally, the outcome condition is a dichotomous representation of intellectual property lock-in, where $LI\{1\}$ indicates a case is locked in and $LI\{0\}$ indicates a case is not locked in. This research will study the cases where lock-in was avoided (expressed as: $1 - LI$ or verbally as "avoidance of lock-in"").

	Inputs			Outcome
	Technical Architecture	IP Architecture	Unlimited Rights	Lock-in
$Case_1$				
$\vdots$	$TA\{0 \leq x_i \leq 1\}$[1]	$IP\{0 \leq x_i \leq 1\}$[2]	$UR\{0,1\}$[3]	$LI\{0,1\}$[4]
$Case_n$				

[1] Indicates membership in set of Quality Technical Architecture
[2] Indicates membership in Set of Accessible IP Architecture
[3] $UR\{1\}$: Unlimited Rights and $UR\{0\}$: Not Unlimited Rights
[4] $LI\{1\}$: Locked-in and $LI\{0\}$: Not Locked-in

Table 1: Design of fsQCA Methodology

Case Selection

An intermediate-N ($10 < n < 20$) fsQCA case study design was chosen with a mix of qualitative and quantitative methods. An intermediate-N was chosen due to the breadth of available DoD software cases. The single or small-N case approach was ruled out because no single type of architecture or intellectual property is common across the entire DoD portfolio. It would be difficult, if not impossible, to identify a single critical case or small number of cases that covered all possibilities. The analysis of such an extreme case would most likely not yield generalizable results. Additionally, and most compelling, given the QCA research design proposed above, Marx and Dusa (2011) found that at least 12 cases were needed to secure a safe threshold against fitting an explanatory outcome to random data.

Sampling Methodology

Comparative researchers argue for a classical theoretical sampling methodology—Most Different, Similar Outcome (MDSO) (Berg-Schlosser & De Meur, 2009; Ragin, 1987, 1994). Ultimately, the MDSO approach was adopted because it offered the greatest potential to yield generalizable results. The sampling methodology used herein was a two-step process. The first step used a convenience sample to collect as many DoD software code bases as possible. The only criteria used in this convenience sample was that the code base had to be part of a DoD contract. In total, 37 cases were collected in the convenience sample. After collecting a sufficiently large convenience sample, the architectures and intellectual property architectures were analyzed. The second step theoretically sampled the collected cases to select those that were unique and/or illustrative of the postulated relationship between architecture and intellectual property. Prior to theoretically sampling for illustrative phenomena, cases with less than 100 files, or classified, were removed from the sample. Additionally, up to three different software versions were sampled for each case (reflected in the number of cases in the convenience sample); however, only the most recent version was used in the ultimate theoretical sample. Table 2 contains a list of the final cases and summary metrics for each case.

System	Size (files)	Dependencies		LOC[1]	SLOC[2]
		Direct	Indirect		
N	24,740	140,055	15,578,401	8,851,927	4,297,613
A	21,879	189,377	154,670,489	8,254,156	5,670,841
B	18,899	323,267	6,742,880	4,530,622	2,783,818
C	6,361	52,274	2,883,437	1,257,869	685,889
D	4,247	18,153	320,172	1,276,114	970,597
E	3,912	13,929	119,016	6,693,882	6,111,764
F	2,869	15,271	317,044	879,115	589,276
G	1,688	10,642	199,571	820,781	399,731
H	1,497	6,860	42,205	253,000	206,579
M	1,466	11,639	896,023	261,023	162,880
I	1,464	6,243	102,614	3,280,227	3,131,633
J	737	3,155	25,185	188,579	119,212
K	497	3,567	107,180	428,623	232,328
L	374	1,525	4,871	69,490	41,783
Totals	90,630	795,957	182,009,088	37,045,408	25,403,944

[1] Lines of Code (LOC)
[2] Source Lines of Code (SLOC)

Table 2: Description of Selected Cases

A critical aspect to an effective MDSO methodology is homogeneity of other explanatory conditions that are not under study. The choice to keep the case selection to DoD cases only recognizes and conforms to the tenets of MDSO. Remaining within the already established DoD program boundaries pares down the number of cases and ensures regulatory, as well as statutory, homogeneity among cases. Furthermore, the selected programs are from commensurate levels of technology maturity (i.e., not comparisons between programs in R&D against programs in production).

Qualitative Methods

Qualitative data collection methods empower theory building and increase exposure to alternate hypotheses. The primary means of qualitative data collection was semistructured guide approach interviews (Patton, 1990). Subjects were interviewed from both engineering, management, and legal roles within selected DoD organizations, as well as from defense contractors for an external perspective.

Each interview was tailored to the characteristics of the case. Interviews used a semistructured guide approach to navigate the contextual details of each program, but a list of interview questions was not used. The choice not to use defined interview questions is a reflection of the contextual variance in each case. Only three questions were uniformly asked across all cases:

1. Are you familiar with the concept of intellectual property lock-in?
2. Are the software program managers or engineers associated with your program experiencing intellectual property lock-in?

3. Does the program office have an unlimited right to the software code?

The answers to these questions served as the basis for the unlimited right input condition and the outcome condition in the fsQCA model. If a program manager or engineer answered “yes” to question two, the case was coded as $LI\{1\}$ or $LI\{0\}$ if the answer was “no”. In all cases, the program managers were familiar with the concept of intellectual property lock-in. Similarly, an answer “yes” to question three was coded as $UR\{1\}$ or $UR\{0\}$ for a “no” answer. If the interviewee was unsure of the answer to the question, a snowball sample was taken to identify individuals who would know the answer (e.g., contracting officer, chief engineer, etc.). Using the snowball sample, interviews continued until an individual could address the questions.

Quantitative Methods

This section focuses on processes of quantitative inquiry that are relevant to the validity of the research design and specific steps taken to reduce threats to validity. Although the basic framework for quantitative analysis used herein involves seven steps, the first two steps are data collection with the remainder being data manipulation (MacCormack et al., 2006; MacCormack et al., 2007):

1. Capture a network representation of software source code.
2. Capture all copyright information from software using regular expressions.
3. Find all the paths (direct and indirect) among files in the network.
4. Calculate visibility scores to each file.
5. Organize files into one of four canonical groups based on visibility scores.
6. Visualize the architecture.
7. Visualize the overlay of copyright data on the architecture.

Following the method outlined by Sturtevant (2013), a network representation of each case was captured using the SciTool's Understand™ static code analysis tool through a Python Application Programming Interface. Since the majority of data is collected in this step (all files and dependencies), a validation check was used to ensure accuracy. Each potential code base was scanned using two different computers. This additional extraction was done to make sure the parameters in the static code analysis tool were properly configured, thus reducing the likelihood of an instrumentation error. In addition to the system-level metrics (i.e., number of files and connections between files), six file-level metrics were observed and an additional five were computed (Table 3).

Variable	Description
	Observed Variables
Filename	Location of file within directory structure
Language	Language of code in file (e.g. C++, C#, Java, Python, etc.)
Fan-In	Number of incoming direct dependencies
Fan-Out	Number of outgoing direct dependencies
LOC	Lines of Code
SLOC	Source Lines of Code
	Computed Variables
Visibility Fan-In	Number of incoming direct and indirect dependencies
Visibility Fan-Out	Number of outgoing direct and indirect dependencies
Cyclic Group ID	A rank order of cyclic group size for which file is a member
Cyclic Group Size	Number of files in cycle group which file is a member
Arch Type	Architectural location of file (i.e. Core, Control, Shared, Periphery)
	Copyright Variables
Owner	Name of copyright holder(s)
License	Type of copyright license(s) (i.e. BSD, MIT, GPL, etc.), if provided
Open Source	Binary classification
Year	Year(s) of copyright

Table 3: File-Level Variable Descriptions

After quantitative data collection concluded for a specific case, the results were compiled and shown to the software programmer or engineer responsible for the case. Feedback was solicited on the accuracy of the observed variables and copyright variables. This informal qualitative triangulation methodology ensured that the file-level metrics mirrored programmer perceptions of the system and were essential in identifying instrumentation errors early in the development of the quantitative analysis framework.

The copyright information is extracted using the process outlined by Berardi (2017), which employs regular expressions to mine for copyright statements in software code. To build and fine-tune the copyright extraction program, a code base was scanned and then a random sample of files was hand-coded to identify false negatives and false positives. Based on the results of the hand coding, changes were made to the regular expressions to reduce false negatives and positives. This iteration cycle continued until no classification errors were found in the random sample of files. The same iterative procedure was applied to three different code bases and iterated until no false negative or false positives were identified in the random sample. The copyright extraction collection process produces four file-level variables (see Table 3 for a description).

Operationalization of Variables

The ability to overlap architectural views allows for combinatorial classification of systems into architecture patterns. An architecture pattern is defined herein as a class of architectures that adheres to specified properties. Using the two extraction methods described previously results in two distinct architectures: intellectual property architecture and technical architecture. Each of these distinct architectures is broken down into subpatterns.

Technical Architecture

In the case of technical architecture, Baldwin et al. (2014) outlined three distinct architectural patterns: core-periphery, multicore, and hierarchical. However, in practice, multicore architectures are relatively uncommon. Therefore, the subpatterns of technical architecture are redefined as: hierarchical architecture, core-periphery architecture, and degraded architecture. The only new category is degraded architecture. A technical architecture is deemed degraded when the core size represents a preponderance of the system, wherein a single cyclic group accounts for more than 50% of the software. In cases with such large cores, the ability to make meaningful impacts to the quality of the software is hampered (Sturtevant, 2013). The theoretical cut points between hierarchical and core-periphery were adopted from Baldwin et al. (2014). However, there is not an academically supported cut line between core-periphery and degraded; 50% is used herein because it represents an obvious degraded architecture. One could make the argument to decrease this number to anything above 30%; however, oftentimes the complexity of architecture is a trade-off for performance. Some practical applications, however, require a high core size to obtain performance; the threshold of 50% allows ample tolerance to account for those cases as core-periphery instead of degraded.

Intellectual Property Architecture

Once all intellectual property characteristics are extracted from software and parsed, the intellectual property information is then overlaid onto a core-periphery view of the software (Berardi et al., 2016). Herein, the combination of intellectual property information and the core-periphery view is referred to as the intellectual property architecture.

The intellectual property architecture is divided into two subpatterns: Accessible and Closed. The unit of analysis for pattern identification is the percentage of files with the highest Visibility Fan-In (VFI) score or hardness-to-kill that are open source or where the copyright holder is the U.S. Government (Berardi et al., 2016). In core-periphery and degraded architecture, these groups of files are defined as the core and shared. However, in the case of hierarchical systems, with very small cores, this measurement is meaningless. Instead, groups of files with high VFI are identified by the deciles of files sorted by VFI. The ninth decile (D9) represents the group of files with the top 10% VFI score and the eighth decile (D8) represents the group of files with the top 20% of VFI scores.

The properties were chosen for an accessible intellectual property architecture because in these patterns, enough code is either owned by the DoD or in the public domain to potentially utilize one of the many competitive techniques outlined by Nash, Schooner, and O'Brien-DeBakey (2013). Thus, these patterns are termed accessible as opposed to open. The other intellectual property pattern is closed architecture; this pattern of architecture is referred to as closed because it is unlikely that enough of the original code is available to utilize one of the competitive techniques by Nash et al. (2013).

Analysis

Analysis using any QCA or set theoretic approach is organized around identifying set relationships and drawing inferences from set relationships across many cases. A condition (**X**), or combination of conditions (**XZ**), is said to be sufficient if it leads to the outcome understudy (**Y**). In set theory, this relationship implies that **X** is a subset of **Y** (i.e., $\boldsymbol{X} \subset \boldsymbol{Y}$). Due to the nature of equifinality and in all likelihood, multiple conditions or groups of conditions are sufficient for a particular outcome. Understanding the relationships between sets allows for inferences about situations that are required for an outcome and situations that may lead to an outcome.

Sufficient Conditions

The goal of sufficient conditions is to identify groupings of causal combinations that lead to the outcome. These groupings of causal combinations will be whole or partial subsets of the outcome. Consequently, for a condition to be sufficient in a fuzzy-set, all the fuzzy-set membership scores (X_i) must be less than or equal to fuzzy-set membership in the outcome (Y_i) (i.e., $X_i \leq Y_i$).

To identify sufficient conditions requires construction of a truth table. The function of a truth table is to categorize the membership level of each case across the range of 8 ($2^3 = 8$) corners of the three-dimensional vector space created by a 3-condition fsQCA. Fuzzy membership in the vector space is calculated by taking the logical "AND" of all possible conditions. In Boolean and Fuzzy algebra, a membership score for AND combinations is calculated as the minimum value across the combinations (i.e., $A * B * C = min(A, B, C)$). The results of these calculations are outlined in Table 4, which shows each case's fuzzy membership in all eight corners of the vector space. [Note: Negation of a condition is indicated using lower case letters: $1 - IP = ip$).] All cases have greater than a 0.5 membership in only one corner of the vector space; these values are highlighted in bold typeface (Table 4). These bold values indicate which corner of the vector space of which each case is a member (e.g., corner ta*ip*UR has four cases with a membership score above 0.5: M, B, G, and J). Therefore, cases M, B, G, and J are said to be a member of set ta*ip*UR.

	taipur	taipUR	taIPur	taIPUR	TAipur	TAipUR	TAIPur	TAIPUR
A	**0.97**	0.00	0.03	0.00	0.00	0.00	0.00	0.00
D	0.19	0.00	0.31	0.00	0.19	0.00	**0.69**	0.00
L	0.00	0.02	0.00	0.04	0.00	0.02	0.00	**0.96**
M	0.00	**0.98**	0.00	0.02	0.00	0.01	0.00	0.01
B	0.00	**0.61**	0.00	0.20	0.00	0.39	0.00	0.20
N	0.00	0.27	0.00	0.27	0.00	**0.68**	0.00	0.32
E	0.00	0.07	0.00	0.03	0.00	**0.93**	0.00	0.03
H	0.00	0.11	0.00	0.02	0.00	**0.89**	0.00	0.02
G	0.00	**0.66**	0.00	0.03	0.00	0.34	0.00	0.03
I	**0.56**	0.00	0.03	0.00	0.44	0.00	0.03	0.00
K	**0.97**	0.00	0.03	0.00	0.00	0.00	0.00	0.00
F	0.00	0.19	0.00	0.03	0.00	**0.81**	0.00	0.03
C	0.06	0.00	**0.60**	0.00	0.06	0.00	0.40	0.00
J	0.00	**0.71**	0.00	0.03	0.00	0.29	0.00	0.03

Table 4: Fuzzy Set of Membership Scores

Membership Scores)"
From the values in Table 4, a truth table is tabulated (Table 5). The truth table for 1-LI shows five distinct combinations of causal conditions that lead to the outcome. One causal combination which leads to LI and two causal combinations for which there are no cases (denoted with a "?"). Because each case is a fuzzy member of each causal combination, the consistency measure is important. If a causal combination demonstrated low consistency across the cases, it would indicate that no agreement exists among the cases on the outcome, or that insufficient evidence emerged from that specific group of causal conditions to determine the outcome clearly and decisively. The threshold used to determine membership in 1-LI was a Con_S score greater than 0.95.

	TA	IP	UR	1-LI	n	Con_S	Cases
1	0	0	0	0	3	0.29	A,I,K
2	0	0	1	1	4	1	M,B,G,J
3	0	1	0	1	1	0.95	C
4	0	1	1	?	0	-	
5	1	0	0	?	0	-	
6	1	0	1	1	4	1	N,E,H,F
7	1	1	0	1	1	0.99	D
8	1	1	1	1	1	1	L

Table 5: Truth Table for Outcome 1-LI

The truth table reveals five possible causal combinations that lead to Not Locked In—see Equation (1). Although this is a complete solution for possibly sufficient conditions, the equation is unnecessarily complex and contains redundant descriptions of the same sample spaces.

$$\mathrm{ta} * \mathrm{ip} * \mathrm{UR} + \mathrm{ta} * \mathrm{IP} * \mathrm{ur} + \mathrm{TA} * \mathrm{ip} * \mathrm{UR} + \mathrm{TA} * \mathrm{IP} * \mathrm{ur} + \mathrm{TA} * \mathrm{IP} * \mathrm{UR} \rightarrow 1 - \mathrm{LI} \quad (1)$$

To reduce the complexity of the equation, Boolean minimization is employed, specifically the enhanced Quine-McCluskey algorithm (Dusa, 2008). The result of the minimization is two possible solutions: Equations 2 and 3.

$$\mathrm{ip} * \mathrm{UR} + \mathrm{IP} * \mathrm{ur} + \mathrm{TA} * \mathrm{UR} \rightarrow 1 - \mathrm{LI} \qquad (2)$$

$$\mathrm{ip} * \mathrm{UR} + \mathrm{IP} * \mathrm{ur} + \mathrm{TA} * \mathrm{IP} \rightarrow 1 - \mathrm{LI} \qquad (3)$$

	Con_S	Cov_S	Cov_U	Cov_U(Eq 8.9)	Cov_U(Eq 8.10)	Cases
ip*UR	1.00	0.61	0.25	0.25	0.56	M,B,G,J; N,E,H,F
IP*ur	0.97	0.15	0.06	0.15	0.06	C; D
TA*IP	1.00	0.23	0.00		0.08	D; L
TA*UR	1.00	0.45	0.01	0.08		N,E,H,F; L
Eq 8.9	0.99	0.84				
Eq 8.10	0.99	0.84				

Table 6: Parameters of Fit for Sufficient Conditions

A less complex expression is reached, which is logically equivalent to the original expression and the information contained in the truth table. In minimizing the complex solution at Equation (1), the nature of causal conditions is reduced to a *conservative expression* of the sufficient causal conditions. The minimization results in two possible solutions: Equations (2) and (3). It is important to note that the two equations reached above are logically equivalent expressions of the same sample space.

Discussion

Previous analysis revealed two expressions for sufficient conditions to avoid lock-in. The conservative expression contained four prime implicants. The term *prime implicants* is used to describe each conjunctive combination in the conservative expressions. Essential prime implicants imply fundamental products that are not covered by any other prime implicants. The prime implicants for the conservative expression were ip*UR, IP*ur, TA*IP, and TA*UR. Two of these prime implicants are said to be *essential prime implicants* (ip*UR and IP*UR) and two are said to be *inessential prime implicants* (TA*IP and TA*UR).

The concept of essential prime implicants is easiest explained using Venn Diagrams (Figures 1a, 1b). In these Venn Diagrams, Figure 1a shows the sample space of the two essential prime implicants, which represent 10 out of 12 cases. Each essential prime implicant covers a unique group of cases, whereas Figure 1b shows the inessential prime implicants, which cover mostly redundant cases. The only case unique to the inessential prime implicants is Case L, which is covered by both inessential prime implicants. Therefore, the addition of TA*IP or TA*UR only adds a single case to the coverage of sufficient conditions; and the inessential prime implicants are interchangeable given that they both add coverage for the same case. These relationships are also succinctly displayed in Table 7, which shows the coverage of each prime implicant. This examination of prime implicants is empirical proof that both conservative expressions are logically equivalent expressions of the same solution.

Findings

The four prime implicants (ip*UR, IP*ur, TA*IP, and TA*UR) identified are all sufficient conditions to avoid lock-in. Each of these paths offers an empirically proven combination of conditions that was successful, in one or more previous cases, at avoiding lock-in. The first sufficient condition, ip*UR, is a somewhat obvious finding. The outcome ip*UR →1-LI is not an experimental or socio-technical phenomena, but rather one directed by statute. Given this directed outcome, the sufficient condition ip*UR is not considered a finding of this research. Although not a direct finding, the appearance of ip*UR, in both the conservative expressions, is a good triangulation of the fsQCA model specification. Had ip*UR not been present, it would have indicated a fsQCA model with a poor theoretical fit.

The second sufficient condition, identified, IP*ur, is a direct finding of the research herein. At the outset of this investigation, intellectual property architecture was postulated as a mechanism of action for intellectual property locked in. Specifically, postulating that those systems with a highly accessible intellectual property architecture would avoid lock-in. The sufficient condition, IP*ur, provides empirical, but not statistically significant, confirmation for this theory.

Accessible intellectual property architecture, absent of any other explanatory conditions, is a sufficient condition, ergo, a mechanism of action for the avoidance of lock-in. Much of the DoD literature on "Open Architectures" argued the advantages of using open source code as well as open standards and theorized it would avoid lock-in, but failed to empirically prove there was a connection. The finding herein is the first time the outcome was proven empirically; thus, this finding is neither obvious nor trivial.

Finding: Accessible intellectual property architecture, absent unlimited rights, is an empirically and theoretically sufficient condition for the avoidance of lock-in.
Although discussed in terms of inessential prime implicants, both TA*IP and TA*UR are sufficient conditions to avoid lock-in, albeit partially redundant. The inclusion of condition TA in both sets is important because it acknowledges the role high-quality technical architecture may contribute to the avoidance of lock-in. Systems with both high levels of modularity and hierarchy are easier to maintain and have less complexity.

Practical Relevance of Sufficient Conditions

The above finding has a direct impact on practice. The empirical proof supports the premise that accessible intellectual property architecture, absent an unlimited right, is a sufficient condition to avoid lock-in, and offers portfolio and program managers new options when confronted with the task of long-term planning. Previously, managers were confined to discussing long-term acquisition strategy options largely in terms of classes of rights (e.g., a manager for a large system would have to foresee the rights the program needed over the life cycle to ensure future competition). Consequently, the acquisition of rights was often thought of as the only way to avoid lock-in. The above finding does not countermand this previous guidance, but instead adds viable options for the manager to achieve the same desired outcome, but with a different approach.

Managers are now capable of taking preventive actions to avoid lock-in without securing any additional rights to intellectual property. Prescriptively, managers that seek to avoid lock-in, without securing additional rights to intellectual property, may choose to formulate system requirements in a way to require or incentivize the use of open source or U.S. Government-owned code in certain parts of the software. As an example, managers may use prescriptive contract language such as the following:

1. The core size of the software (measured in file size) shall not exceed 10% of the entire system (measured in file size).
2. The core of the software should contain greater than 50% open source software.

The first requirement above is an example of a threshold requirement (i.e., compliance is mandatory); whereas, the second requirement is an example of an objective requirement (i.e., compliance is optional, but oftentimes incentivized). The introduction of objective requirements formulated towards creating an accessible intellectual property architecture, coupled with an incentive structure, provides a means to both objectively measure the outcome and incentivize the creation of an architecture to avoid lock-in. For example, a contractor could be incentivized by offering $1 thousand for every tenth of a percentage point if the core size is below 10% or $50 thousand for every percentage point increase in open source above 50% of the core. This would imply a 1% reduction in core size and would net a contractor $100 thousand. Prior to this research, no means existed to objectively measure the accessibility of software in a meaningful way for contract language; nor was there a way to specify an architectural location for the desired open source software. The contract language exemplified above provides both an objective measure and the ability to specify relative location.

It is important to note that contractors are under no mandatory requirement that they will accept the incentive or work towards the incentivized goal. Although these types of requirements are advantageous for the DoD, a contractor may conduct a cost-benefit analysis on the revenue accrued as a result of incentives compared to the long-term revenue accrued as a result of creating a system without an accessible intellectual property architecture. Such a cost-benefit analysis may favor creation of a system that ignores accessible intellectual property architecture and potentially yields lock-in. Depending on the incentive amounts and life cycle of a given system, this may be a rational economic choice. To avoid this situation, further research is needed into the optimal quantity of economic incentives, which would give contractors adequate incentive to build systems with accessible intellectual property architectures.

Care must also be taken to ensure the requirements outlined above do not create the wrong incentive (e.g., a contractor could pollute a code base with unnecessary open source code). While this increased level of open source may increase the likelihood of avoiding lock-in, it may also impact performance of the system. Consequently, any use of the above contracting requirements must be employed in conjunction with a rigorous performance testing scheme to ensure no system degradation results. Although polluting a code base with open source code to achieve an incentive is a real threat, it is partially mitigated by specifying the relative location in the core. Which files will end up in the core is not always obvious to programmers when building the code. In fact, it is often an emergent property based on the incomprehensible indirect dependencies. Understanding the full set of indirect dependencies in a piece of software is beyond human cognition; therefore, accurately predicting which files will end up in the core is also a difficult task for humans. The inability to foresee and predict which files will be core or shared offers some mitigation against open source pollution or gaming incentives.

Potential Criticisms and Limitations

The operationalization of the set UR is a point of certain criticism from DoD legal experts. Many different paths lead to secure membership in set UR; however, for this research set UR was operationalized as the purposeful acquisition of an unlimited right. This set did not include cases that received an unlimited right due to expiration of a restricted right, mismanagement on the part of the contractor, or any other method that did not directly acquire an unlimited right as a condition to, or a deliverable of, a contract. Operationalization of this type narrows membership in set UR and ignores some cases that may otherwise have an unlimited right. This choice for operationalization sought to intentionally exclude those cases that obtained an unlimited right by accident. Cases A and K were examples of such a scenario. These cases obtained an unlimited right arguably due to contractor mismanagement, not foresight or planning of the program office. Future research may identify an improved method to capture the set of cases that purposefully acquired an unlimited right and exclude those that did not, but at the time of publication, this was the most effective and logical operationalization of set UR.

Limitations are those factors of the research that may impact the generalizability of the research results. The first threat to generalizability is the number of cases present in some of the sets of conditions. Specifically, sets ta*IP*ur, TA*IP*ur, and TA*IP*UR are only covered by a single case. With only one case present, it is difficult to argue that these are unique results and not a byproduct of one or more threats to external validity or a byproduct of randomness in data. In QCA research, each set of conditions would ideally be supported by multiple cases (n) and only those sets with a number of cases above a selected threshold would be factored into the results (e.g., $n > 5$ or $n > 10$). If five or 10 cases all demonstrate the same configuration, the argument that these results are random becomes more difficult, as the same configuration allows for more control against randomness in the data.

Future Research

The authors found general avenues of investigation, as revealed by the results of this research, which would improve the validity of the findings herein. More cases are needed with outcome locked in, followed by an exhaustive investigation into the specific conditions or sets of conditions that specifically yield locked in. Furthermore, any future fsQCA approaches would benefit from an improved gradient scale measure of lock-in that is generalizable across the DoD or, ideally, across all software systems. This gradient scale could be an operationalization of the current spectrum of rights outlined in the DFARS or a completely novel approach.

Conclusions

Intellectual property lock-in is a wicked problem, particularly pervasive in the DoD and other monopsonies. This work identified that little research exists on the mechanisms of action that induce intellectual property lock-in. It further postulates the conjuncture of architecture and intellectual property is one such mechanism of action.

To investigate links between intellectual property, architecture, and lock-in, an intermediate-N fuzzy-set Qualitative Comparative Analysis research approach was formulated and executed using 14 DoD software cases representing over 34 million lines of code. The model used three input conditions: high-quality technical architecture, accessible intellectual property architecture, and unlimited rights to study the avoidance of lock-in.

This research finds within the sample, an intellectual property architecture, absent of any other explanatory conditions, is sufficient to avoid lock-in. Suggesting that software architectures with small core groups (suggested value 10% of total lines of code) are more robust to lock-in, particularly when coupled with open source or software to which the DoD has an unlimited license comprising more than 50% of the intellectual property in the core.

Understanding the list of conditions that are sufficient to avoid lock-in helps both portfolio and program managers refine the sample of options available when conducting long-term planning. This does not imply that engineers and system designers must now become experts in the legal domain, but that choices they make are part of a larger scheme of multiple conjunctural causality that could eventually lead to lock-in.

References

Anderson, W. C. (2003). Comparative analysis of intellectual property issues relating to the acquisition of commercial and noncommercial items by the federal government. *Public Contract Law Journal*, *33*(1), 37–61.

Baker, K. (2011). Intellectual property clauses to watch for in government contracts. *Contract Management*, *52*(5), 50–55.

Baker, J. M., Haque, S., Mccarthy, J. E., & Raddock, A. (2015). Government data rights challenges: Tips on how to avoid & respond. *Contract Management*, *55*(11), 20–28.

Baldwin, C. Y. (2015). *Bottlenecks, modules and dynamic architectural capabilities* (Harvard Business School Working Paper Series No. 15-028). Retrieved from https://www.hbs.edu/faculty/Publication%20Files/15-028_ec57f2ea-890d-4243-b9f3-60784359bab9.pdf

Baldwin, C. Y., & Henkel, J. (2014). *Modularity and intellectual property protection* (Harvard Business School Working Paper Series No. 14-046). Retrieved from https://www.hbs.edu/faculty/Pages/item.aspx?num=45969

Baldwin, C. Y., MacCormack, A., & Rusnak, J. (2014). Hidden structure: Using network methods to map system architecture. *Research Policy, 43*(8), 1381–1397.

Berardi, C. W. (2017). *Intellectual property and architecture: How architecture influences intellectual property lock-in* (Doctoral dissertation). Retrieved from http://hdl.handle.net/1721.1/112005

Berardi, C. W., Cameron, B., & Crawley, E. (2017). Informing policy through quantification of the intellectual property lock-in associated with DoD acquisition. *Defense Acquisition Research Journal*, *24*(3), 432–467.

Berardi, C. W., Cameron, B., Sturtevant, D., Baldwin, C. Y., & Crawley, E. (2016). Architecting out software intellectual property lock-In: A method to advance the efficacy of BBP. In *Proceedings of the 13th Annual Acquisition Research Symposium* (pp. 184–201). Monterey, CA: Naval Postgraduate School.

Berg-Schlosser, D., & De Meur, G. (2009). Comperative research design. In C. C. Ragin & B. Rihoux (Eds.), *Configurational comparative methods* (pp. 19–32). Los Angeles: SAGE Publications.

Berg-Schlosser, D., De Meur, G., Rihoux, B., & Ragin, C. C. (2009). Qualitative Comperative Analysis (QCA) as an approach. In B. Rihoux & C. C. Ragin (Eds.), *Configurational comparative methods* (pp. 1–19). Los Angeles: SAGE Publications.

Cronqvist, L., & Berg-Schlosser, D. (2009). Multi-value QCA (mvQCA). In C. C. Ragin & B. Rihoux (Eds.), *Configurational comparative methods* (pp. 69–86). Los Angeles: SAGE Publications.

Dix, N. O., Lavallee, F. A., & Welch, K. C. (2003). Fear and loathing of federal contracting: Are commercial companies really afraid to do business with the federal government? Should they be? *Public Contract Law Journal*, *33*(1), 5–36.

Duberstein, D. (1988). Validation of proprietary data restrictions : How contractors can protect their rights in technical data against government challenge. *National Contract Management Journal*, *22*(1), 25–32.

Dungan, C. P. (2010). Less is more: Encouraging greater competition in computer software procurement by simplifying the DFARS licensing scheme. *Public Contract Law Journal*, *39*(3), 465–482.

Dusa, A. (2008). A mathematical approach to the Boolean minimization problem. *Quality & Quantity*, *44*(1), 99.

Henkel, J., Baldwin, C. Y., & Shih, W. C. (2012). *IP modularity: Profiting from innovation by aligning product architecture with intellectual property* (Harvard Business School Working Paper Series No. 13-012). Retrieved from https://hbswk.hbs.edu/item/ip-modularity-profiting-from-innovation-by-aligning-product-architecture-with-intellectual-property

Hinrichs, M. R. M. (1967). Proprietary data and trade secrets under Department of Defense contracts. *Military Law Review*, *36*, 61–90.

Jacobides, M. G., Knudsen, T., & Augier, M. (2006). Benefiting from innovation: Value creation, value appropriation and the role of industry architectures. *Research Policy*, *35*(8 SPEC. ISS.), 1200–1221.

Jacobides, M. G., & Tae, J. C. (2015). Kingpins, bottlenecks, and value dynamics along a sector. *Organization Science*, *26*(3), 889–907.

Jacobides, M. G. (2013, April), Velosa, F. M., & Wolter, C. *Ripples through the value chain: How upstream innovation shapes profit and scope in a sector.* Paper presented at the 2013 Wharton Technology and Innovation Conference, Philadelphia, PA.

Langlois, R. N. (2002). Modularity in technology and organization. *Journal of Economic Behavior and Organization*, *49*(1), 19–37.

Larson, D. (2007). Yesterday's technology, tomorrow: How the government's treatment of intellectual property prevents soldiers from receiving the best tools to complete their mission. *The John Marshall Review of Intellectual Property Law*, 171–201.

Lavenue, L. M. (1998). Technical data rights in government procurement: Intellectual property rights in computer software and the indicia of information systems and information technology. *University of San Francisco Law Review*, *32*(1), 1–75.

Lazure, A. C., & Church, J. H. (1954). The shadow and substance of intellectual property in Defense Department research and development contracts. *The Federal Bar Journal*, *14*, 300–320.

Liedke, E. J., & Simonis, J. D. (2014). *Increasing competitive actions: A focus on technical data rights associated with non-commercial hardware items*. Monterey, CA: Naval Postgraduate School.

Lill, B. (2009). Restrictive legends in federal procurement: Is the risk of losing data rights too great? *Public Contract Law Journal*, *38*(4), 895–911.

MacCormack, A., Rusnak, J., & Baldwin, C. Y. (2006). Exploring the structure of complex software designs: An empirical study of open source and proprietary code. *Management Science*, *52*(7), 1015–1030.

MacCormack, A., Rusnak, J., & Baldwin, C. Y. (2007). *The impact of component* modularity *on design evolution: Evidence from the software industry* (Harvard Business School Working Paper Series No. 8–38). Retrieved from https://www.hbs.edu/faculty/Publication%20Files/08-038_187f1243-7d1e-464a-bddd-bf7c09873284.pdf

Madden, T. J., Marvin, C. R., Jr., & Reams, J. T. (2006). The role of independent research and development funds in securing contractor rights to intellectual property. *Contract Management*, *46*(7), 6–12.

Marx, A., & Dusa, A. (2011). Crisp-Set Qualitative Comparative Analysis (csQCA), contradictions and consistency benchmarks for model specification. *Methodological Innovations Online*, *6*(2), 103–148.

Mazour, E. (2009). If you fix it, they will come: Drawing lessons learned from patents for dealing with rights in technical data. *Public Contract Law Journal*, *38*(3), 667–688.

Murray, R. (2012). *Intellectual property and technical data rights: It's about the money*. U.S. Army War College Civilian Research Project. Retrieved from http://www.dtic.mil/dtic/tr/fulltext/u2/a593245.pdf

Muzzelo, L., & Arndt, C. (2014). Data rights for science and technology projects. *Defense Acquisition Research Journal*, *21*(2), 625–650.

Nash, R. C., Schooner, S. L., & O'Brien-DeBakey, K. R. (2013). *The government contracts reference book: A comprehensive guide to the language of procurement* (4th ed.). Riverwoods, IL: Wolters Kluwer Law & Business.

Pages, K. (2013). Rights in technical data and computer software – A tenuous balance between the government and the contractor. *Contract Management*, *53*(2), 42–50.

Patton, M. Q. (1990). *Qualitative evaluation and research methods* (2nd ed.). Newbury Park, CA: SAGE Publications.

Pickarz, J. (2015). Evaluating intellectual property and data rights in competition source selections—Leveraging the "assertions process" to a new level to foster open systems architecture. In *Proceedings of the Twelfth Annual Acquisition Research Symposium*. Monterey, CA: Naval Postgraduate School.

Pisano, G. P., & Teece, D. J. (2007). How to capture value from innovation: Shaping intellectual property and industry architecture. *California Management Review*, *50*(1), 278–296.

Ragin, C. C. (1987). *The comparative method: Moving beyond qualitative and quantitative strategies*. Berkeley, CA: University of California Press.

Ragin, C. C. (1994). *Constructing social research*. Thousand Oaks, CA: Pine Oaks Press.

Ragin, C. C. (2005). Core versus tangential assumptions in comparative research. *Studies in Comparative International Development*, *40*(1), 33–38.

Ragin, C. C. (2006). Set relations in social research: Evaluating their consistency and coverage. *Political Analysis*, *14*(3), 291–310.

Ragin, C. C. (2009). Qualitative Comparative Analysis using fuzzy sets (fsQCA). In B. Rihoux & C. C. Ragin (Eds.), *Configurational comparative methods* (pp. 87–121). Los Angeles: SAGE Publications.

Rihoux, B. (2003). Bridging the gap between the qualitative and quantitative worlds? A retrospective and prospective view on Qualitative Comparative Analysis. *Field Methods*, *15*(4), 351–365.

Rihoux, B., & De Meur, G. (2009). crisp-set Qualitative Comparative Analysis (csQCA). In C. C. Ragin & B. Rihoux (Eds.), *Configurational comparative methods* (pp. 33–68). Los Angeles: Sage Publications.

Rihoux, B., & Lobe, B. (2009). The case for Qualitative Comparative Analysis (QCA): Adding leverage for thick cross-case comparison. In C. C. Ragin & D. Byrne (Eds.), *The SAGE handbook of case-based methods* (pp. 222–242). Thousand Oaks, CA and London: SAGE Publications.

Rittel, H. W. J., & Webber, M. M. (1973). Dilemmas in a general theory of planning. *Policy Sciences*, *4*(2), 155–169.

Sharp, G. S. (2003). A layman's guide to intellectual property in defense contracts. *Public Contract Law Journal*, *33*(1), 99–137.

Sidebottom, D. M. (2003). Intellectual property in federal government contracts: The past, the present, and one possible future. *Public Contract Law Journal*, *33*(1), 63–97.

Simchak, M. S. (2003). Protecting rights in technical data and computer software: Applying the ten practical rules and their corollaries. *Public Contract Law Journal*, *33*(1), 139–162.

Simchak, M. S., & Vogel, D. A. (1994). A few words of advice: Protecting intellectual property when contracting with the Department of Defense according to the October 1988 regulations. *Public Contract Law Journal*, *23*(2), 141–167.

Simon, H. A. (1962). The architecture of complexity. *Proceedings of the American Philosophical Society*, *106*(6), 33–42.

Stucke, M. E. (2013). Is competition always good? *Journal of Antitrust Enforcement*, *1*(1), 162–197.

Sturtevant, D. (2013). *System design and the cost of architectural complexity*. Cambridge, MA: Massachusetts Institute of Technology.

Trend, C. C. (2005). Killing the goose that laid the golden egg: Data rights law and policy in Department of Defense contracts. *Public Contract Law Journal*, *34*(2), 287–336.

Ulrich, K. (1995). The role of product architecture in the manufacturing firm. *Research Policy*, *24*(3), 419–440.

Varone, F. F., Rihoux, B., & Marx, A. (2006). A new method for policy evaluation? Longstanding challenges and the possibilities of Qualitative Comparative Analysis (QCA). In B. Rihoux & H. Grimm (Eds.), *Innovative comparative methods for policy analysis: Beyond the quantitative-qualitative divide* (pp. 213–236). Boston, MA: Springer.

Endnotes

[1]Pub. L. No. 112-81 § 815, 125 Stat. 1491 (2011)

[2]Pub. L. No. 113-291 § 801, 128 Stat. 3425 (2014)

[3]The Defense Acquisition University offers courses on intellectual property, but completion of these courses is not mandatory and the material is not of sufficient detail to cover the entirety ofregulation and statute.

[4]A Justification and Authorization is a contracting artifact used to justify why a contract was not competitively procured.

Author Biographies

Maj Chris Berardi, USAF, is currently a program manager at the Air Force Rapid Capabilities Office. Prior to beginning at the RCO, he served as both an acquisition and intelligence officer, and was a Fellow in the Chief of Staff of the Air Force Captains Prestigious PhD Program. Maj Berardi received his undergraduate degree from the United States Air Force Academy, a master's degree in Engineering Management, and a PhD in Engineering Systems from Massachusetts Institute of Technology (MIT).

(E-mail address: christopher.berardi@us.af.mil)

Dr. Bruce Cameron is the Director of the System Architecture Lab at MIT. His research interests include technology strategy, system architecture, and the management of product platforms. Dr. Cameron is also a co-founder of Technology Strategy Partners (TSP), a consultancy focused on supporting the CTO's office in the management of R&D. He is a past board member of the University of Toronto. Dr. Cameron received his undergraduate degree from the University of Toronto, a master's degrees in Aeronautics & Aeronautics and in Technology Policy from MIT, and a PhD in Engineering Systems from MIT.

(E-mail address: bcameron@mit.edu)